

HUAWEI NetEngine 8100 X/NetEngine 8000 X/ NetEngine 8000E X series V800R023C00SPC500

Configuration Guide

Issue 01
Date 2023-09-30



HUAWEI TECHNOLOGIES CO., LTD.



Copyright © Huawei Technologies Co., Ltd. 2023. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
 Bantian, Longgang
 Shenzhen 518129
 People's Republic of China

Website: <https://www.huawei.com>

Email: support@huawei.com

Contents

1 Configuration.....	1
1.1 IP Routing.....	1
1.1.1 Basic IP Routing Configuration.....	1
1.1.1.1 Basic IP Routing Description.....	1
1.1.1.1.1 Overview of Basic IP Routing.....	1
1.1.1.1.2 Understanding IP Routing.....	1
1.1.1.1.3 Application Scenarios for IP Routing.....	25
1.1.1.1.4 Appendix List of Port Numbers of Common Protocols.....	28
1.1.1.1.5 Terminology for IP Routing.....	29
1.1.1.2 Basic IP Routing Configuration.....	30
1.1.1.2.1 Overview of Basic IP Routing.....	30
1.1.1.2.2 Configuration Precautions for Basic IP Routing.....	31
1.1.1.2.3 Configuring the Router ID.....	31
1.1.1.2.4 Configuring IPv4 Multi-Topology.....	32
1.1.1.2.5 Configuring IPv6 Multi-Topology.....	33
1.1.1.2.6 Configuring IPv4 FRR.....	33
1.1.1.2.7 Configuring IPv6 FRR.....	37
1.1.1.2.8 Configuring Route Recursion to the Default Route.....	40
1.1.1.2.9 Configuring the Device to Advertise Host Routes of Directly Connected Interfaces.....	41
1.1.1.2.10 Configuring Association Between Direct Routes and a VRRP Group.....	42
1.1.1.2.11 Configuring an IPv4 Direct Route to Respond to L3VE Interface Status Changes After a Delay.....	45
1.1.1.2.12 Configuring an IPv6 Direct Route to Respond to L3VE Interface Status Changes After a Delay.....	48
1.1.1.2.13 Configuring the Association Between the IPv4 Direct Route and PW Status.....	50
1.1.1.2.14 Configuring the Association Between the IPv6 Direct Route and PW Status.....	52
1.1.1.2.15 Configuring AGGs to Load Balance Downstream Traffic.....	55
1.1.1.2.16 Configuring the Advertisement of Public Network IPv4 ARP Vlink Direct Routes.....	57
1.1.1.2.17 Configuring the Advertisement of IPv6 NDP Vlink Direct Routes on the Public Network.....	59
1.1.1.2.18 Configuring the Device to Filter ARP or ND Entries When Advertising VPN Vlink Direct Routes.....	61
1.1.1.2.19 Configuring the Association Between IPv4 Direct Routes and IPsec Instance Status.....	63
1.1.1.2.20 Configuring Route or Tunnel Recursion Suppression in Case of Flapping.....	65
1.1.1.2.21 Maintaining IP Routes.....	66
1.1.1.2.22 Configuration Examples for IP Routing Basis.....	71

1.1.2 IPv4 Static Route Configuration.....	102
1.1.2.1 Static Routes Description.....	102
1.1.2.1.1 Overview of Static Routes.....	102
1.1.2.1.2 Understanding Static Routes.....	102
1.1.2.2 IPv4 Static Route Configuration.....	114
1.1.2.2.1 Overview of IPv4 Static Routes.....	114
1.1.2.2.2 Configuration Precautions for IPv4 static route.....	115
1.1.2.2.3 Configuring IPv4 Static Routes.....	118
1.1.2.2.4 Configuring IPv4 Floating Static Routes.....	124
1.1.2.2.5 Configuring Association between LDP and Static Routes.....	125
1.1.2.2.6 Configuring IPv4 Static Route Detection.....	128
1.1.2.2.7 Configuring FRR for IPv4 Static Routes.....	139
1.1.2.2.8 Configuration Examples for IPv4 Static Routes.....	140
1.1.3 IPv6 Static Route Configuration.....	184
1.1.3.1 IPv6 Static Route Configuration.....	184
1.1.3.1.1 Overview of IPv6 Static Routes.....	184
1.1.3.1.2 Configuration Precautions for IPv6 Static Route.....	184
1.1.3.1.3 Configuring IPv6 Static Routes.....	184
1.1.3.1.4 Configuring IPv6 Floating Static Routes.....	189
1.1.3.1.5 Configuring IPv6 Static Route Detection.....	190
1.1.3.1.6 Configuring FRR for IPv6 Static Routes.....	199
1.1.3.1.7 Configuration Examples for IPv6 Static Routes.....	201
1.1.4 OSPF Configuration.....	238
1.1.4.1 OSPF Description.....	239
1.1.4.1.1 Overview of OSPF.....	239
1.1.4.1.2 Understanding OSPF.....	241
1.1.4.2 OSPF Configuration.....	332
1.1.4.2.1 Overview of OSPF.....	332
1.1.4.2.2 Configuration Precautions for OSPF.....	335
1.1.4.2.3 Configuring Basic OSPF Functions.....	342
1.1.4.2.4 Configuring OSPF on the NBMA or P2MP Network.....	354
1.1.4.2.5 Adjusting OSPF Route Selection.....	360
1.1.4.2.6 Controlling OSPF Routing Information.....	369
1.1.4.2.7 Adjusting the OSPF Network Convergence Speed.....	385
1.1.4.2.8 Configuring OSPF Delay Advertisement.....	396
1.1.4.2.9 Configuring OSPF Neighbor Relationship Flapping Suppression.....	398
1.1.4.2.10 Disabling OSPF Interface Flapping Suppression.....	401
1.1.4.2.11 Configuring Routing Loop Detection for Routes Imported into OSPF.....	402
1.1.4.2.12 Configuring OSPF Flush Source Tracing.....	402
1.1.4.2.13 Configuring an OSPF Hostname.....	404
1.1.4.2.14 Configuring an OSPF Stub Area.....	405
1.1.4.2.15 Configuring an NSSA.....	407

1.1.4.2.16 Configuring OSPF Local MT.....	410
1.1.4.2.17 Configuring an OSPF Sham Link.....	412
1.1.4.2.18 Configuring BFD for OSPF.....	415
1.1.4.2.19 Configuring OSPF IP FRR.....	421
1.1.4.2.20 Configuring OSPF GR Helper.....	427
1.1.4.2.21 Improving OSPF Network Security.....	429
1.1.4.2.22 Configuring the Network Management Function of OSPF.....	433
1.1.4.2.23 Configuring Whitelist Session-CAR for OSPF.....	434
1.1.4.2.24 Configuring Micro-Isolation CAR for OSPF.....	435
1.1.4.2.25 Configuring the Threshold Alarm Function for the Number of OSPF Neighbors.....	436
1.1.4.2.26 Maintaining OSPF.....	436
1.1.4.2.27 Configuring OSPF Multi-Area Adjacency.....	439
1.1.4.2.28 Configuration Examples for OSPF.....	451
1.1.5 OSPFv3 Configuration.....	545
1.1.5.1 OSPFv3 Description.....	545
1.1.5.1.1 Introduction to OSPFv3.....	545
1.1.5.1.2 Understanding OSPFv3.....	546
1.1.5.2 OSPFv3 Configuration.....	608
1.1.5.2.1 OSPFv3 Overview.....	608
1.1.5.2.2 Configuration Precautions for OSPFv3.....	609
1.1.5.2.3 Configuring Basic OSPFv3 Functions.....	610
1.1.5.2.4 Configuring OSPFv3 Attributes on Different Types of Networks.....	618
1.1.5.2.5 Adjusting OSPFv3 Route Selection.....	622
1.1.5.2.6 Controlling OSPFv3 Routing Information.....	626
1.1.5.2.7 Configuring OSPFv3 Delay Advertisement.....	639
1.1.5.2.8 Disabling DN Bit Setting in OSPFv3 LSAs.....	641
1.1.5.2.9 Configuring OSPFv3 Neighbor Relationship Flapping Suppression.....	643
1.1.5.2.10 Configuring Routing Loop Detection for Routes Imported to OSPFv3.....	646
1.1.5.2.11 Disabling OSPFv3 Interface Flapping Suppression.....	647
1.1.5.2.12 Configuring OSPFv3 Flush Source Tracing.....	647
1.1.5.2.13 Configuring an OSPFv3 Hostname.....	649
1.1.5.2.14 Configuring an OSPFv3 Stub Area.....	650
1.1.5.2.15 Configuring an OSPFv3 NSSA.....	652
1.1.5.2.16 Configuring OSPFv3 IP FRR.....	654
1.1.5.2.17 Configuring BFD for OSPFv3.....	658
1.1.5.2.18 Configuring OSPFv3 Authentication.....	663
1.1.5.2.19 Configuring OSPFv3 Fast Convergence.....	672
1.1.5.2.20 Configuring the OSPFv3 GR Helper.....	678
1.1.5.2.21 Configuring OSPFv3 to Report Network Topology Information to BGP-LS.....	680
1.1.5.2.22 Improving OSPFv3 Network Stability.....	680
1.1.5.2.23 Configuring the Network Management Function of OSPFv3.....	685
1.1.5.2.24 Whitelist Session-CAR for OSPFv3.....	686

1.1.5.2.25 Configuring Micro-Isolation CAR for OSPFv3.....	687
1.1.5.2.26 Configuring the Threshold Alarm Function for the Number of OSPFv3 Neighbors.....	688
1.1.5.2.27 Maintaining OSPFv3.....	688
1.1.5.2.28 Configuration Examples for OSPFv3.....	691
1.1.6 RIP Configuration.....	749
1.1.6.1 RIP Description.....	749
1.1.6.1.1 Overview of RIP.....	749
1.1.6.1.2 Understanding RIP.....	750
1.1.6.2 RIP Configuration.....	759
1.1.6.2.1 Overview of RIP.....	759
1.1.6.2.2 Configuration Precautions for RIP.....	760
1.1.6.2.3 Configuring Basic RIP Functions.....	760
1.1.6.2.4 Preventing Routing Loops.....	766
1.1.6.2.5 Adjusting RIP Route Selection.....	769
1.1.6.2.6 Controlling RIP Routing Information.....	774
1.1.6.2.7 Configuring RIP Fast Convergence.....	782
1.1.6.2.8 Configuring Dynamic BFD for RIP.....	787
1.1.6.2.9 Configuring Static BFD for RIP.....	790
1.1.6.2.10 Configuring OPS to Control Packet Sending and Receiving on RIP Interfaces.....	791
1.1.6.2.11 Improving the RIP Network Security.....	793
1.1.6.2.12 Configuring RIP Network Management.....	803
1.1.6.2.13 Maintaining RIP.....	803
1.1.6.2.14 Configuration Examples for RIP.....	805
1.1.7 RIPng Configuration.....	828
1.1.7.1 RIPng Description.....	828
1.1.7.1.1 Overview of RIPng.....	828
1.1.7.1.2 Understanding RIPng.....	829
1.1.7.2 RIPng Configuration.....	835
1.1.7.2.1 Overview of RIPng.....	835
1.1.7.2.2 Configuration Precautions for RIPng.....	836
1.1.7.2.3 Configuring Basic RIPng Functions.....	836
1.1.7.2.4 Controlling the Sending and Receiving of RIPng Packets.....	839
1.1.7.2.5 Preventing Routing Loops.....	841
1.1.7.2.6 Adjusting RIPng Route Selection.....	844
1.1.7.2.7 Controlling RIPng Routing Information.....	847
1.1.7.2.8 Configuring RIPng Timers.....	853
1.1.7.2.9 Configuring IPsec Authentication for RIPng.....	854
1.1.7.2.10 Maintaining RIPng.....	856
1.1.7.2.11 RIPng Configuration Examples.....	857
1.1.8 IS-IS Configuration.....	866
1.1.8.1 IS-IS Description.....	866
1.1.8.1.1 Overview of IS-IS.....	866

1.1.8.1.2 Understanding IS-IS.....	866
1.1.8.1.3 Application Scenarios for IS-IS.....	940
1.1.8.2 IS-IS Configuration.....	941
1.1.8.2.1 Overview of IS-IS.....	942
1.1.8.2.2 Configuration Precautions for IS-IS.....	942
1.1.8.2.3 Configuring Basic IPv4 IS-IS Functions.....	945
1.1.8.2.4 Configuring IPv4 IS-IS Route Selection.....	959
1.1.8.2.5 Configuring IPv4 IS-IS to Interact with Other Routing Protocols.....	971
1.1.8.2.6 Adjusting the IPv4 IS-IS Route Convergence Speed.....	976
1.1.8.2.7 Configuring an IS-IS Multi-Instance Process.....	986
1.1.8.2.8 Enabling the Advertisement of IPv4 Delay Information.....	988
1.1.8.2.9 Configuring Interface MSD Advertisement (IPv4).....	989
1.1.8.2.10 Configuring an IS-IS Process to Advertise IPv4 Packet Loss Rates	990
1.1.8.2.11 Configuring LFA Link Attribute Advertisement.....	992
1.1.8.2.12 Configuring IS-IS Extended Prefix Attribute Advertisement (IPv4).....	993
1.1.8.2.13 Configuring Static BFD for IS-IS.....	994
1.1.8.2.14 Configuring Dynamic BFD for IS-IS.....	996
1.1.8.2.15 Configuring Track BFD for IS-IS.....	1000
1.1.8.2.16 Configuring IS-IS IPv4 MT to Isolate Multicast Services from Unicast Services.....	1001
1.1.8.2.17 Configuring IPv4 IS-IS Route Summarization.....	1005
1.1.8.2.18 Configuring IS-IS Auto FRR.....	1006
1.1.8.2.19 Configuring Basic IPv6 IS-IS Functions.....	1009
1.1.8.2.20 Configuring IPv6 IS-IS Route Selection.....	1020
1.1.8.2.21 Configuring IPv6 IS-IS to Interact with Other Routing Protocols.....	1032
1.1.8.2.22 Adjusting the IPv6 IS-IS Route Convergence Speed.....	1037
1.1.8.2.23 Enabling the Advertisement of IPv6 Delay Information.....	1047
1.1.8.2.24 Configuring an IS-IS Process to Advertise IPv6 Packet Loss Rates	1049
1.1.8.2.25 Configuring IS-IS Extended Prefix Attribute Advertisement (IPv6).....	1050
1.1.8.2.26 Configuring IS-IS Neighbor Relationship Flapping Suppression.....	1052
1.1.8.2.27 Disabling IS-IS Interface Flapping Suppression.....	1054
1.1.8.2.28 Configuring Routing Loop Detection for Routes Imported into IS-IS.....	1054
1.1.8.2.29 Configuring IS-IS Purge Source Tracing.....	1055
1.1.8.2.30 Configuring Dynamic BFD for IPv6 IS-IS.....	1056
1.1.8.2.31 Configuring Track BFD for IPv6 IS-IS.....	1060
1.1.8.2.32 Configuring IS-IS IPv6 MT to Isolate Multicast Services from Unicast Services.....	1061
1.1.8.2.33 Configuring IS-IS MT to Isolate IPv4 Services from IPv6 Services.....	1065
1.1.8.2.34 Configuring IPv6 IS-IS Route Summarization.....	1069
1.1.8.2.35 Configuring IPv6 IS-IS Auto FRR.....	1070
1.1.8.2.36 Configuring an IS-IS Link Group.....	1073
1.1.8.2.37 Configuring IS-IS Local MT.....	1075
1.1.8.2.38 Improving IS-IS Network Security.....	1078
1.1.8.2.39 Configuring Whitelist Session-CAR for IS-IS.....	1084

1.1.8.2.40 Configuring Micro-Isolation CAR for IS-IS.....	1085
1.1.8.2.41 Maintaining IS-IS.....	1085
1.1.8.2.42 IS-IS Configuration Examples.....	1089
1.1.9 BGP Configuration.....	1172
1.1.9.1 BGP Description.....	1172
1.1.9.1.1 Overview of BGP.....	1172
1.1.9.1.2 Understanding BGP.....	1175
1.1.9.2 BGP Configuration.....	1303
1.1.9.2.1 BGP Overview.....	1303
1.1.9.2.2 Configuration Precautions for BGP.....	1305
1.1.9.2.3 Configuring Basic BGP Functions.....	1317
1.1.9.2.4 Configuring the Format of BGP 4-Byte AS Numbers.....	1325
1.1.9.2.5 Configuring BGP Route Attributes.....	1326
1.1.9.2.6 Configuring BGP Routing Policies.....	1344
1.1.9.2.7 Configuring BGP XPL.....	1367
1.1.9.2.8 Configuring BGP Route Summarization.....	1370
1.1.9.2.9 Configuring BGP to Generate a Summary Default Route.....	1372
1.1.9.2.10 Configuring a BGP Peer Group.....	1373
1.1.9.2.11 Configuring Distributed BGP Peers.....	1377
1.1.9.2.12 Configuring BGP Parallelization.....	1377
1.1.9.2.13 Configuring a BGP Route Reflector.....	1378
1.1.9.2.14 Configuring a BGP Confederation.....	1384
1.1.9.2.15 Configuring BGP Community Attributes.....	1385
1.1.9.2.16 Configuring the BGP Large-Community Attribute.....	1390
1.1.9.2.17 Configuring Prefix-based BGP ORF.....	1392
1.1.9.2.18 Adjusting the BGP Network Convergence Speed.....	1393
1.1.9.2.19 Configuring a Dynamic BGP Peer Group.....	1402
1.1.9.2.20 Configuring a BGP Device to Send a Default Route to Its Peer.....	1403
1.1.9.2.21 Configuring a Device to Advertise BGP Supernet Unicast Routes to BGP Peers.....	1405
1.1.9.2.22 Configuring BGP Load Balancing.....	1406
1.1.9.2.23 Configuring BGP LSP Load Balancing.....	1413
1.1.9.2.24 Configuring a BGP SR LSP.....	1414
1.1.9.2.25 Configuring Path MTU Auto Discovery.....	1418
1.1.9.2.26 Configuring BGP Route Recursion to the Default Route.....	1421
1.1.9.2.27 Configuring BGP Next Hop Recursion Based on a Route-Policy.....	1422
1.1.9.2.28 Configuring AIGP value on a Route-Policy.....	1423
1.1.9.2.29 Configuring BGP Add-Path.....	1427
1.1.9.2.30 Configuring the POPGO Function.....	1429
1.1.9.2.31 Configuring the Device to Perform the Label Pop-go Action for Packets Carrying the Label Added to Each Received Non-labeled Unicast Route.....	1430
1.1.9.2.32 Configuring conversion from BGP IPv4 Unicast Routes to Labeled Routes.....	1432
1.1.9.2.33 Configuring BFD for BGP.....	1434
1.1.9.2.34 Configuring BGP Peer Tracking.....	1439

1.1.9.2.35 Configuring BGP Auto FRR.....	1440
1.1.9.2.36 Configuring Delayed Response to BGP Next Hop Recursion Changes.....	1441
1.1.9.2.37 Setting a Specified BGP Peer or Each Peer in a Peer Group as an Independent Update Peer-Group.....	1444
1.1.9.2.38 Configuring a Delay in Releasing Obtained Labels in a BGP LSP FRR Switchover Scenario.....	1445
1.1.9.2.39 Configuring the Route Server Function.....	1447
1.1.9.2.40 Configuring the BGP GR Helper.....	1448
1.1.9.2.41 Enabling GR for BGP Peers.....	1449
1.1.9.2.42 Configuring BGP G-Shut.....	1451
1.1.9.2.43 Configuring BGP Best-external.....	1452
1.1.9.2.44 Configuring BMP.....	1453
1.1.9.2.45 Configuring BGP Route Dampening.....	1458
1.1.9.2.46 Configuring Suppression on BGP Peer Flapping.....	1460
1.1.9.2.47 Configuring Flapping Suppression Involved in BGP Next Hop Recursion.....	1461
1.1.9.2.48 Configuring BGP-LS.....	1462
1.1.9.2.49 Configuring the Entropy Label Capability for a BGP LSP.....	1466
1.1.9.2.50 Configuring BGP RPD.....	1468
1.1.9.2.51 Configuring the Capability of Creating MPLS Local IFNET Tunnels for BGP Peers.....	1470
1.1.9.2.52 Configuring the YANG Management Mode of BGP.....	1472
1.1.9.2.53 Improving BGP Security.....	1474
1.1.9.2.54 Configuring BGP Extensions.....	1489
1.1.9.2.55 Configuring BGP Multi-Instance.....	1490
1.1.9.2.56 Maintaining BGP.....	1491
1.1.9.2.57 BGP Route Selection Rules.....	1501
1.1.9.2.58 Configuration Examples for BGP.....	1580
1.1.10 BGP4+ Configuration.....	1740
1.1.10.1 BGP4+ Configuration.....	1740
1.1.10.1.1 Overview of BGP4+.....	1740
1.1.10.1.2 Configuration Precautions for BGP4+.....	1743
1.1.10.1.3 Configuring Basic BGP4+ Functions.....	1743
1.1.10.1.4 Controlling Route Advertisement.....	1750
1.1.10.1.5 Controlling BGP4+ Route Selection.....	1756
1.1.10.1.6 Configuring BGP4+ Routing Policies.....	1768
1.1.10.1.7 Configuring BGP4+ XPL.....	1786
1.1.10.1.8 Configuring BGP4+ Route Recursion to the Default Route.....	1788
1.1.10.1.9 Configuring BGP4+ to Generate Locator Routes.....	1789
1.1.10.1.10 Configuring BGP4+ Load Balancing.....	1790
1.1.10.1.11 Configuring BGP4+ to Generate a Summary Default Route.....	1796
1.1.10.1.12 Configuring BGP4+ Connection Parameters.....	1797
1.1.10.1.13 Configuring BGP4+ RR.....	1802
1.1.10.1.14 Configuring a BGP4+ Confederation.....	1808
1.1.10.1.15 Configuring BGP4+ Community Attributes.....	1809
1.1.10.1.16 Configuring Prefix-based BGP4+ ORF.....	1814

1.1.10.1.17 Configuring the BGP4+ Add-Path Function.....	1815
1.1.10.1.18 Configuring BFD for BGP4+.....	1817
1.1.10.1.19 Configuring BGP4+ Auto FRR.....	1819
1.1.10.1.20 Setting a Specified BGP4+ Peer or Each Peer in a Peer Group as an Independent Update Peer-Group.....	1820
1.1.10.1.21 Configuring the Route Server Function.....	1821
1.1.10.1.22 Configuring the BGP4+ GR Helper.....	1822
1.1.10.1.23 Configuring BMP.....	1823
1.1.10.1.24 Enabling GR for BGP4+ Peers.....	1827
1.1.10.1.25 Configuring BGP4+ Route Dampening.....	1828
1.1.10.1.26 Configuring Suppression on BGP4+ Peer Flapping.....	1830
1.1.10.1.27 Configuring Flapping Suppression Involved in BGP4+ Next Hop Recursion.....	1831
1.1.10.1.28 Configuring BGP-LS (IPv6).....	1832
1.1.10.1.29 Configuring BGP SAVNET (IPv6).....	1836
1.1.10.1.30 Configuring BGP4+ Security.....	1840
1.1.10.1.31 Configuring BGP4+ 6PE.....	1853
1.1.10.1.32 Enabling the Device to Recurse BGP IPv6 Unicast Routes to LSPs.....	1858
1.1.10.1.33 Maintaining BGP4+.....	1860
1.1.10.1.34 Configuration Examples for BGP4+.....	1866
1.1.11 Routing Policy Configuration.....	1893
1.1.11.1 Routing Policy Description.....	1893
1.1.11.1.1 Overview of Routing Policy.....	1893
1.1.11.1.2 Understanding Routing Policies.....	1895
1.1.11.1.3 Application Scenarios for Routing Policies.....	1910
1.1.11.2 Routing Policy Configuration.....	1913
1.1.11.2.1 Overview of Routing Policies.....	1913
1.1.11.2.2 Configuration Precautions for Routing Policy.....	1915
1.1.11.2.3 Configuring an IP Prefix List.....	1915
1.1.11.2.4 Configuring an ACL.....	1917
1.1.11.2.5 Configuring an AS_Path Filter.....	1919
1.1.11.2.6 Configuring a Community Filter.....	1920
1.1.11.2.7 Configuring a Large-Community Filter.....	1921
1.1.11.2.8 Configuring an Extcommunity Filter.....	1922
1.1.11.2.9 Configuring an RD Filter.....	1924
1.1.11.2.10 Configuring a Route-Policy.....	1924
1.1.11.2.11 Applying Filters to Received Routes.....	1958
1.1.11.2.12 Applying Filters to Routes to Be Advertised.....	1964
1.1.11.2.13 Applying Filters to Imported Routes.....	1970
1.1.11.2.14 Configuration Examples for Routing Policies.....	1974
1.1.12 XPL Configuration.....	2009
1.1.12.1 XPL Description.....	2009
1.1.12.1.1 Overview of XPL.....	2009
1.1.12.1.2 XPL Fundamentals.....	2010

1.1.12.1.3 Application Scenarios for XPL.....	2043
1.1.12.2 XPL Configuration.....	2044
1.1.12.2.1 Overview of XPL.....	2044
1.1.12.2.2 Configuration Precautions for XPL.....	2046
1.1.12.2.3 Introduction to XPL Paragraph-by-Paragraph Editing.....	2047
1.1.12.2.4 Configuring a Global Variable Set.....	2059
1.1.12.2.5 Configuring a Route Attribute Set.....	2061
1.1.12.2.6 Configuring a Route-Filter.....	2085
1.1.12.2.7 Applying Route-Filters.....	2089
1.1.12.2.8 Configuration Examples for XPL.....	2092
1.1.12.2.9 XPL Paragraph Editing Clauses.....	2112
1.1.13 Route Monitoring Group Configuration.....	2207
1.1.13.1 Route Monitoring Group Description.....	2207
1.1.13.1.1 Overview of Route Monitoring Groups.....	2207
1.1.13.1.2 Understanding Route Monitoring Groups.....	2207
1.1.13.1.3 Application Scenarios for Route Monitoring Groups.....	2209
1.1.13.1.4 Terminology for Route Monitoring Group.....	2211
1.1.13.2 Route Monitoring Group Configuration.....	2211
1.1.13.2.1 Overview of Route Monitoring Groups.....	2211
1.1.13.2.2 Configuration Precautions for Route Monitoring Group.....	2212
1.1.13.2.3 Configuring a Route Monitoring Group.....	2212
1.1.13.2.4 Configuration Examples for Route Monitoring Groups.....	2217

1 Configuration

1.1 IP Routing

NOTE

Properly plan route configurations to prevent network problems caused by route loops. For details, see Layer 3 Routing Loop Prevention for Routers ([Carrier users](#)/[Enterprise users](#)).

1.1.1 Basic IP Routing Configuration

1.1.1.1 Basic IP Routing Description

1.1.1.1.1 Overview of Basic IP Routing

Definition

As a basic concept on data communication networks, routing is the process of packet relaying or forwarding, and the process provides route information for packet forwarding.

Purpose

During data forwarding, routers, routing tables, and routing protocols are indispensable. Routing protocols are used to discover routes and contribute to the generation of routing tables. Routing tables store the routes discovered by various routing protocols, and routers select routes and implement data forwarding.

1.1.1.1.2 Understanding IP Routing

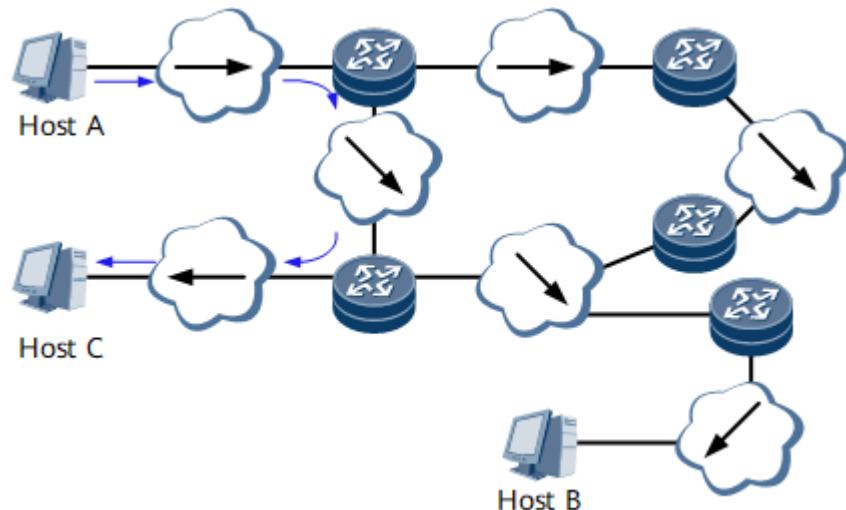
Routers

On the Internet, network connection devices control network traffic and ensure data transmission quality on networks. Common network connection devices include hubs, bridges, switches, and routers.

As a standard network connection device, a router is used to select routes and forward packets. Based on the destination address in the received packet, a router selects a path to send the packet to the next router. The last router is responsible for sending the packet to the destination host. In addition, a router can select an optimal path for data transmission.

For example, in **Figure 1-1**, traffic from Host A to Host C needs to pass through three networks and two routers. The hop count from a router to its directly connected network is zero. The hop count from a router to a network that the router can reach through another router is one. The rest can be deduced by analogy. If a router is connected to another router through a network, a network segment exists between the two routers, and they are considered adjacent on the Internet. In **Figure 1-1**, the bold arrows indicate network segments. The routers do not need to know about the physical link composition of each network segment.

Figure 1-1 Network segment and hop count



Network sizes may vary greatly, and the actual lengths of network segments vary as well. Therefore, you can set a weighted coefficient for the network segments of each network and then measure the cost of a route based on the number of network segments.

A route with the minimal network segments is not necessarily optimal. For example, a route passing through three high-speed Local Area Network (LAN) network segments may be a better choice than one passing through two low-speed Wide Area Network (WAN) network segments.

Routing Protocols

Routing protocols are rules used by routers to discover routes, add routes, and maintain routing tables for packet forwarding.

Routing Tables

A router searches a routing table for routes, and each router maintains at least one routing table.

Routing tables store the routes discovered by various routing protocols. Based on the generation method, routes in a routing table consist of the following types:

- Routes discovered by link layer protocols, which are also called interface routes or direct routes
- Static routes configured by the network administrator
- Dynamic routes that are discovered by dynamic routing protocols

Routing Table Types

Each router maintains a local core routing table, and each routing protocol maintains its own routing table.

- Protocol routing table

A protocol routing table stores routing information discovered by the protocol. A routing protocol can import and advertise routes generated by other routing protocols. For example, if a router that runs Open Shortest Path First (OSPF) needs to use OSPF to advertise direct routes, static routes, or Intermediate System to Intermediate System (IS-IS) routes, the router needs to import these routes into the OSPF routing table.

- Local core routing table

A local core routing table stores protocol routes and optimal routes and selects routes based on the priorities of routing protocols and costs of routes. You can run the **display ip routing-table** command to view the local core routing table of a router.

NOTE

Each router that supports Layer 3 virtual private network (L3VPN) maintains a management routing table (local core routing table) for each VPN instance.

Contents in the Routing Table

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, the **display ip routing-table** command displays brief information about the routing table.

```
<HUAWEI> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table: Public
Destinations : 8      Routes : 8

Destination/Mask Proto Pre Cost Flags NextHop      Interface
          0.0.0.0/0  Static 60  0    D   10.1.4.2    GigabitEthernet1/0/0
          10.1.4.0/30 OSPF   10  0    D   10.1.4.1    GigabitEthernet1/0/0
          10.1.4.1/32 Direct  0   0    D   127.0.0.1  InLoopBack0
          10.1.4.2/32 OSPF   10  0    D   10.1.4.2    GigabitEthernet1/0/0
          127.0.0.0/8 Direct  0   0    D   127.0.0.1  InLoopBack0
          127.0.0.1/32 Direct  0   0    D   127.0.0.1  InLoopBack0
          127.255.255.255/32 Direct 0   0    D   127.0.0.1 InLoopBack0
          255.255.255.255/32 Direct 0   0    D   127.0.0.1 InLoopBack0
```

A routing table contains the following key entries:

- Destination: indicates the destination IP address or the destination network address of an IP packet.

- Mask: indicates the network mask. The network mask and the destination address are used together to identify the address of the network segment where the destination host or router resides.
 - The address of the network segment where the destination host or router resides can be calculated using after the AND operation on the destination address and network mask. For example, if the destination address is 1.1.1.1 and the mask is 255.255.255.0, the address of the network segment where the host or the router resides is 1.1.1.0.
 - The mask, which consists of several consecutive 1s, can be expressed either in dotted decimal notation or by the number of consecutive 1s in the mask. For example, the length of the mask 255.255.255.0 is 24, and therefore, the mask can also be expressed as 24.
- Protocol: indicates the name of a routing protocol.
- Pre: indicates the priority of a route that is added to the IP routing table. If multiple routes have the same destination but different next hops or outbound interfaces or these routes are static routes or discovered by different routing protocols, the one with the highest priority (the smallest value) is selected as the optimal route. For the route priority of each routing protocol, see [Table 1-1](#).
- Cost: indicates the route cost. When multiple routes to the same destination have the same priority, the route with the smallest cost is selected as the optimal route.

NOTE

The Preference is used during the selection of routes discovered by different routing protocols, whereas the Cost is used during the selection of routes discovered by the same routing protocol.

- Flags:
 - Route flag:
 - R: indicates a recursive route.
 - D: indicates a route that is downloaded to the FIB.
 - T: indicates a route whose next hop belongs to a VPN instance.
 - B: indicates a black-hole route.
 - Next hop: indicates the IP address of the next router through which an IP packet passes.
 - Interface: indicates the outbound interface that forwards an IP packet.

Based on the destination addresses, routes can be classified into the following types:

- Network segment route: The destination is a network segment.
- Host route: The destination is a host.

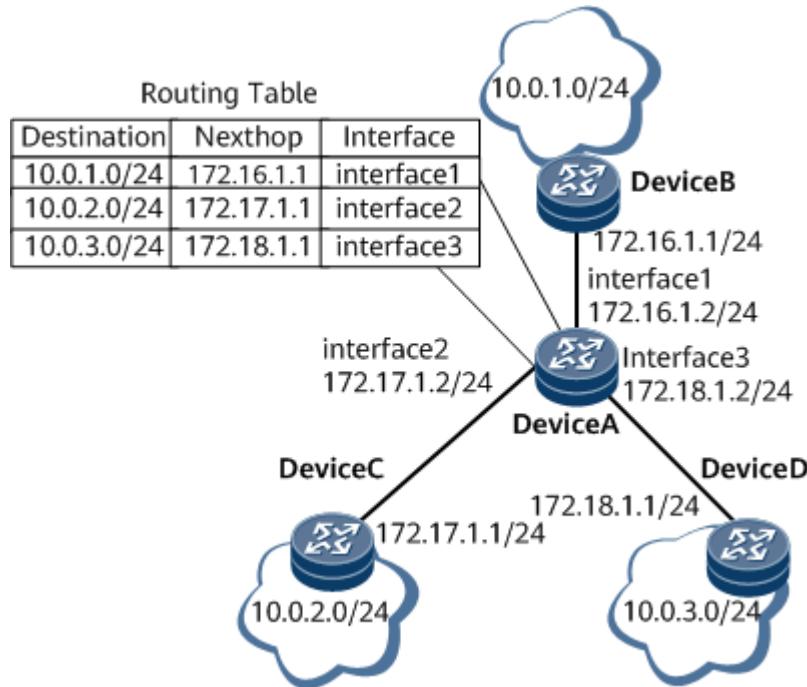
In addition, based on whether the destination is directly connected to the router, route types are as follows:

- Direct route: The router is directly connected to the destination network.
- Indirect route: The router is indirectly connected to the destination network.

Setting a default route can reduce the number of routing entries in the routing table. When a router cannot find a route in the routing table, the router uses the default route (destined for 0.0.0.0/0) to send packets.

In [Figure 1-2](#), Device A is connected to three networks, and therefore, it has three IP addresses and three outbound interfaces. [Figure 1-2](#) shows the routing table on Device A.

Figure 1-2 Routing table



Route Recursion

Routes can be used to forward traffic only when they have directly connected next hops. However, this condition may not be met when routes are generated.

Therefore, the system needs to search for the directly connected next hops and corresponding outbound interfaces, and this process is called route recursion. In most cases, BGP routes, static routes, and UNRs do not have directly connected next hops, and route recursion is required.

For example, the next hop IP address of a BGP route is the IP address of a non-directly connected peer's loopback interface, and therefore, the BGP route needs to perform recursion. Specifically, the system searches the IP routing table for a direct route (IGP route in most cases) that is destined for the next hop IP address of the BGP route and then adds the next hop IP address and outbound interface of the IGP route to the IP routing table to generate a FIB entry.

The next hop IP address of a BGP VPN route is the IP address of a non-directly connected PE's loopback interface, and the BGP route needs to recurse to a tunnel. Specifically, the system searches the tunnel list for a tunnel that is destined for this loopback IP address and then adds the tunnel information to the routing table to generate a FIB entry.

Static and Dynamic Routes

Static routes can be easily configured and have low requirements on the system. They apply to simple, stable, and small-scale networks. However, they cannot automatically adapt to network topology changes. Therefore, static routes require subsequent maintenance.

Dynamic routing protocols have their routing algorithms and can automatically adapt to network topology changes. They apply to the network equipped with a number of Layer 3 devices. Dynamic route configurations are complex. Dynamic routes have higher requirements on a system than static ones do and consume network resources.

Classification of Dynamic Routing Protocols

Dynamic routing protocols can be classified according to the following dimensions:

Based on the Application Scope

Based on the application scope, routing protocols are classified into the following types:

- Interior Gateway Protocols (IGPs): run within an AS. Common IGPs include RIP, OSPF, and IS-IS.
- Exterior Gateway Protocols (EGPs): run between ASs. At present, BGP is the most widely used EGP.

Based on the Routing Algorithm

Based on the algorithm used, routing protocols can be classified into the following types:

- Distance-vector routing protocols: include RIP and BGP. BGP is also called a path-vector protocol.
- Link-state routing protocols: include OSPF and IS-IS.

The main difference between the preceding two algorithms lies in the methods of discovering and calculating routes.

Based on the Destination Address Type

Based on the destination address type, routing protocols are classified into the following types:

- Unicast routing protocols: include RIP, OSPF, BGP, and IS-IS.
- Multicast routing protocols: include Protocol Independent Multicast-Sparse Mode (PIM-SM).

This chapter describes unicast routing protocols. For details about multicast routing protocols, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description - IP Multicast*.

Static routes and dynamic routes discovered by routing protocols are managed in a unified manner. These routes can be imported from each other to implement **Re-advertisement of Routing Information**.

Routing Protocol and Route Priority

Route Priority

Routing protocols (including static route) may discover different routes to the same destination, but not all the routes are optimal. Only one routing protocol is used each time to determine the optimal route to a destination. Routing protocols and static routes have their priorities. When multiple route sources exist, the route with the highest priority (smallest value) is selected as the optimal route. [Table 1-1](#) lists routing protocols and their default priorities.

Value 0 indicates a direct route, and value 255 indicates any route learned from an unreliable source. A smaller value indicates a higher priority.

Table 1-1 Routing protocols and their default priorities

Routing Protocol or Route Type	Routing Priority
Direct	0
OSPF	10
IS-IS	15
Static	60
RIP	100
OSPF ASE	150
OSPF NSSA	150
BGP	255
IBGP	255
EBGP	255

Priorities can be manually configured for routes of routing protocols, except for direct routes. In addition, the priorities of static routes can be different.

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X defines external and internal priorities. The external priorities refer to the priorities set by users for routing protocols. [Table 1-1](#) lists the default external priorities.

When different routing protocols are configured with the same priority, the system selects the optimal route based on the internal priority. For the internal priority of each routing protocol, see [Table 1-2](#).

Table 1-2 Internal priority of routing protocols

Routing Protocol or Route Type	Routing Priority
Direct	0
OSPF inter-area	10
OSPFv3 inter-area	10
IS-IS Level-1	15
IS-IS Level-2	18
EBGP	20
Static	60
UNR	65
RIP	100
RIPng	100
OSPF ASE	150
OSPFv3 ASE	150
OSPF NSSA	150
OSPFv3 NSSA	150
IBGP	200

For example, both an OSPF route and a static route are destined for 10.1.1.0/24, and their protocol priorities are set to 5. In this case, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X selects the optimal route based on the internal priorities listed in **Table 1-2**. The internal priority of OSPF (10) is higher than that of the static route (60). Therefore, the device selects the route discovered by OSPF as the optimal route.

 **NOTE**

- If multiple OSPFv2 processes learn routes to the same destination and the external and internal priorities of the routes are the same, the system selects the route with the smallest link cost; if the link costs of the routes are the same, the routes participate in load balancing. If multiple OSPFv3 processes learn routes to the same destination and the external and internal priorities of the routes are the same, the system selects the route with the smallest process ID.
- If multiple IS-IS processes learn routes to the same destination and the external and internal priorities of the routes are the same, the device selects the route with the smallest link cost; if the link costs of the routes are the same, the routes perform load balancing.
- If multiple RIP/RIPng processes learn routes to the same destination and the external and internal priorities of the routes are the same, the device selects the route with the smallest link cost; if the link costs of the routes are the same, the routes perform load balancing.

Priority-based Route Convergence

Definition

Priority-based route convergence is an important technology to improve network reliability. It provides faster route convergence for key services. For example, when a fault occurs on the network, to shorten the interruption of key services, real-time multicast services require fast convergence of routes to the multicast source. The MPLS VPN transport network requires fast convergence of end-to-end routes between PEs.

Different routes can be set with different convergence priorities, which can be critical, high, medium, and low listed in descending order. Critical is the highest convergence priority; low is the lowest convergence priority. The system performs route convergence based on the convergence priorities and certain convergence rules. That is, route convergence is performed based on a certain scheduling ratio to guide uninterrupted service forwarding.

Purpose

With the network convergence, requirements on service differentiation increase. Carriers require that routes for key services, such as voice over IP (VoIP) and video conferencing services, converge faster than those for common services. Therefore, the system needs to process different routes based on different convergence priorities to improve network reliability.

Route Convergence Priority

Priorities in route convergence are critical, high, medium, and low, which are listed in descending order. [Table 1-3](#) lists the default convergence priorities of public network routes. You can set convergence priorities for routes as needed based on the specific networking.

Table 1-3 Default convergence priorities of public network routes

Routing Protocol or Route Type	Convergence Priority
Direct	critical
Static	medium
OSPF and IS-IS host routes with 32-bit masks	medium
OSPF (except host routes with 32-bit masks)	low
IS-IS (except host routes with 32-bit masks)	low
RIP	low
BGP	low

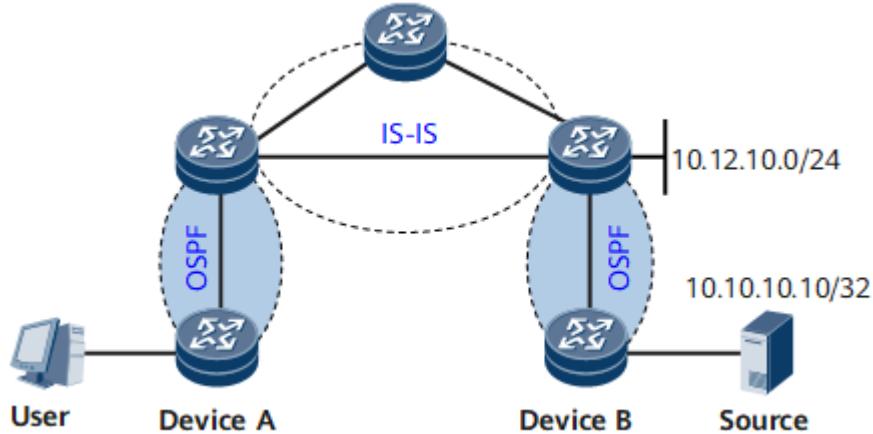
 NOTE

For VPN routes, the convergence priorities of only the OSPF and IS-IS host routes with 32-bit masks are medium, and those of the other routes are low.

Typical Application

IGPs run on the network shown in [Figure 1-3](#), the receiver is connected to Device A, and the multicast source server 10.10.10.10/32 is connected to Device B. It is required that the route to the multicast server converge prior to other routes such as 10.12.10.0/24. In this case, you can set the convergence priority of the route 10.10.10.10/32 to be higher than that of the route 10.12.10.0/24. In this manner, when routes converge on the network, the route 10.10.10.10/32 to the multicast source converges first, ensuring the forwarding of multicast services.

Figure 1-3 Network diagram of priority-based route convergence



Load Balancing and Route Backup

Load Balancing

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports the multi-route model (multiple routes with the same destination and priority). Load balancing can be performed among multiple routes discovered by the same routing protocol if they have the same destination and cost. In each routing protocol view, you can run the **maximum load-balancing number** command to perform load balancing. Load balancing is classified into the following types:

- Per-packet load balancing

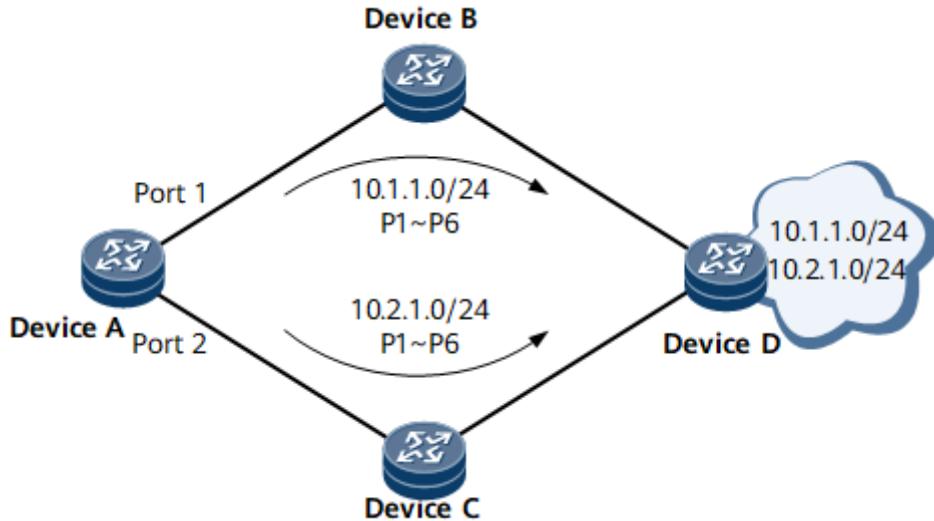
When per-packet load balancing is configured, the router forwards packets destined for the same destination through each path at the IP layer, and each time the next hop address is different from the one selected last time.

- Per-flow load balancing

When per-flow load balancing is configured, the router forwards packets according to the 5-tuple (source IP address, destination IP address, source port number, destination port number, and protocol). If packets have the same 5-

tuple, the router always selects the next hop address that is the same as the one selected last time to send the packets. [Figure 1-4](#) shows the networking.

Figure 1-4 Networking diagram of per-flow load balancing



Device A needs to forward packets to 10.1.1.0/24 and 10.2.1.0/24. In per-flow load balancing, packets of the same flow are always transmitted along the previous path. Device A forwards packets as follows:

- The first packet P1 to 10.1.1.0/24 is forwarded through Port 1, and all subsequent packets to 10.1.1.0/24 are forwarded through Port 1.
- The first packet P1 to 10.2.1.0/24 is forwarded through Port 2, and all subsequent packets to 10.2.1.0/24 are forwarded through Port 2.

Currently, RIP, OSPF, BGP, and IS-IS support load balancing, and static routes also support load balancing.

NOTE

The maximum number of equal-cost routes for load balancing varies with the router model.

Route Backup

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports route backup to improve network reliability. You can configure multiple routes to the same destination as required. The route with the highest priority functions as the primary route, and the other routes with lower priorities function as backup routes.

In most cases, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X uses the primary route to forward packets. If the link of the primary route fails, the primary route becomes inactive. The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X then selects a backup route with the highest priority to forward packets, and the original primary route becomes a backup route. When the original primary route recovers, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X restores and reselects the optimal route. Because the original

primary route has the highest priority, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X selects this route to send packets. Therefore, the backup route becomes the primary route.

Principles of IP FRR

Overview

Fast Reroute (FRR) functions when the lower layer (physical layer or data link layer) detects a fault. The lower layer reports the fault to the upper layer routing system and immediately forwards packets through a backup link.

If a link fails, FRR helps reduce the impact of the link failure on services transmitted on the link.

Background

On traditional IP networks, when a fault occurs at the lower layer of the forwarding link, the physical interface on the router goes Down. After the router detects the fault, it instructs the upper layer routing system to recalculate routes and then update routing information. The routing system takes several seconds to reselect an available route.

For services that are sensitive to packet loss and delay, a convergence time of several seconds is intolerable because it may lead to service interruptions. For example, the maximum convergence time tolerable for Voice over IP (VoIP) services is within milliseconds. IP FRR enables the forwarding system to detect a fault and then to take measures to restore services as soon as possible.

Classification and Implementation

IP FRR, which is designed for routes on IP networks, consists of public network IP FRR and VPN IP FRR.

- Public network IP FRR: protects routers on the public network.
- VPN IP FRR: protects Customer Edges (CEs).



The static routes that are imported between public and private networks do not support IP FRR.

IP FRR is implemented as follows:

- IP FRR can be enabled or disabled using commands.
- When optimal routes are selected from the routes discovered by routing protocols, a backup link is selected for each preferred primary link based on the protocol priority, and then the forwarding information of primary and backup links is provided for the forwarding engine.

Implementation of IP FRR Between Different Protocols

When IP FRR between different protocols is enabled, and optimal routes are selected from protocol routes, a backup link is selected for each preferred primary link based on the protocol priority, and then the forwarding information of primary and backup links is provided for the forwarding engine.

If the forwarding engine detects that the primary link is unavailable after IP FRR between different protocols is enabled, the system can use the backup link to forward traffic before the routes converge on the control plane.

Comparison Between IP FRR and Load Balancing

Table 1-4 Comparison between IP FRR and load balancing

Feature	Description
IP FRR	Implements FRR through a backup route. IP FRR is applicable to networks where a master link and a backup link exist and load balancing is not configured.
Load balancing	Implements fast route switching through equal-cost routes and applies to the multi-link networking with load balancing.

Re-advertisement of Routing Information

Different routing protocols may discover different routes because they adopt different routing algorithms. When the scale of a network is large and multiple routing protocols run on the network, these protocols need to re-advertise their discovered routes.

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, the routes discovered by a routing protocol can be imported into the routing table of another routing protocol. Each protocol has its mechanism to import routes. For details, see "Routing Policy."

Indirect Next Hop

Definition

Indirect next hop is a technique used to speed up route convergence. This technique can change the direct association between route prefixes and next hop information into an indirect association. Indirect next hop allows next hop information to be refreshed independently of the prefixes of the same next hop, which speeds up route convergence.

Purpose

In the scenario requiring route recursion, when IGP routes or tunnels are switched, forwarding entries are rapidly refreshed, which implements fast route convergence and reduces the impact of route or tunnel switching on services.

Mapping Between the Route Prefix and the Next Hop

Mapping between route prefixes and next hops is the basis of indirect next hop. To meet the requirements of route recursion and tunnel recursion in different scenarios, next hop information includes the address family, original next hop address, and tunnel policy. The system assigns an index to each next hop,

performs route recursion, communicates the recursion result to the routing protocol, and then delivers forwarding entries.

On-Demand Route Recursion

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, the route to a reachable address is called a dependent route. The system forwards packets based on dependent routes. The process of finding a dependent route based on the next hop address is called route recursion.

On-demand route recursion indicates that when a dependent route changes, only the next hop associated with the dependent route performs recursion again. If the route destination address is the original next hop address or network segment address of next hop information, any route changes affect the recursion result of the next hop information. Otherwise, route changes do not affect next hop information. Therefore, when a route changes, you can perform recursion again only on the associated next hop by assessing the destination address of the route. For example, if the original next hop address of the route 2.2.2.2/32 is 1.1.1.1, the route that the original next hop 1.1.1.1 depends on may be 1.1.1.1/32 or 1.1.0.0/16. If the route 1.1.1.1/32 or 1.1.0.0/16 changes, the recursion result of the original next hop 1.1.1.1 is affected.

With respect to tunnel recursion, when a tunnel alternates between Up and Down, perform recursion again on the next hop whose original next hop address is the same as the destination address of the tunnel.

Recursion Policy

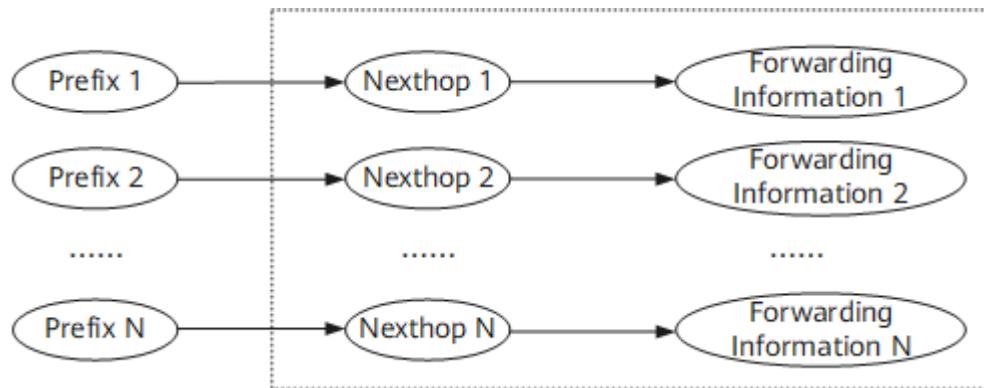
A recursion policy is used to control the recursion result of the next hop to meet requirements of different scenarios. In route recursion, behaviors do not need to be controlled by the recursion policy. Instead, recursion behaviors only need to comply with the longest match rule. In addition, the recursion policy needs to be applied only when VPN routes recurse to tunnels.

By default, the system selects Label Switched Paths (LSPs) for VPNs without performing load balancing. If load balancing or other types of tunnels are required, configure a tunnel policy and bind it to a tunnel. After the tunnel policy is applied, the system uses the tunnel bound to the tunnel policy or selects a tunnel based on the priorities specified in the tunnel policy during next hop recursion.

Mechanism for Indirect Next Hop

Without indirect next hop, the forwarding information corresponds to the prefix, and therefore, the route convergence time is decided by the number of route prefixes. With indirect next hop, multiple route prefixes correspond to one next hop. Forwarding information is added to the forwarding table using the next hop, and traffic with relevant route prefixes can be switched, which speeds up route convergence.

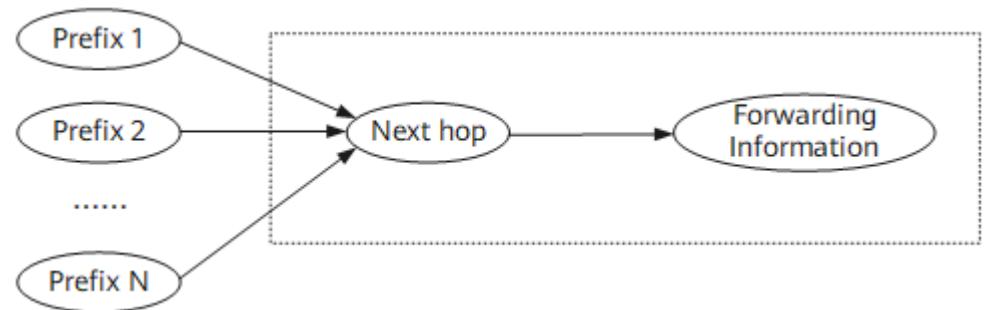
Figure 1-5 Implementation without indirect next hop



As shown in **Figure 1-5**, without indirect next hop, prefixes are totally independent, each corresponding to its next hop and forwarding information. When a dependent route changes, the next hop corresponding to each prefix performs recursion and forwarding information is updated based on the prefix. In this case, the convergence time is decided by the number of prefixes.

Note that prefixes of a BGP peer have the same next hop, forwarding information, and refreshed forwarding information.

Figure 1-6 Implementation with indirect next hop



As shown in **Figure 1-6**, with indirect next hop, prefixes of routes from the same BGP peer share the same next hop. When a dependent route changes, only the shared next hop performs recursion and forwarding information is updated based on the next hop. In this case, routes of all prefixes can converge at a time. Therefore, the convergence time is irrelevant to the number of prefixes.

Comparison Between Route Recursion and Tunnel Recursion

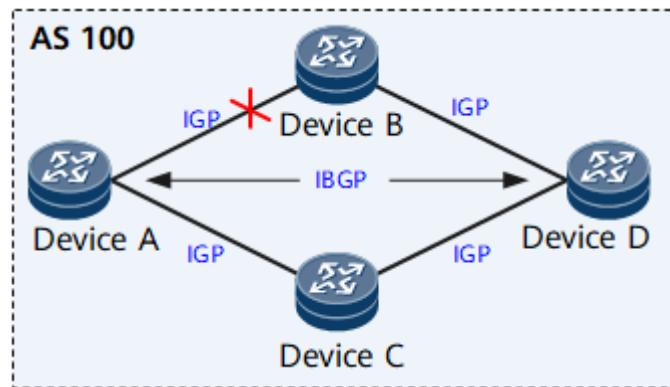
The following table lists differences between route recursion and tunnel recursion.

Table 1-5 Differences between route recursion and tunnel recursion

Recursion Type	Description
Route recursion	<ul style="list-style-type: none"> Applies to BGP public network routes. Is triggered by route changes. Supports next hop recursion based on the specified routing policy.
Tunnel recursion	<ul style="list-style-type: none"> Applies to BGP VPN routes. Is triggered by tunnel or tunnel policy changes. Recursion behaviors can be controlled using a tunnel policy to meet requirements of different scenarios.

IBGP Route Recursion to an IGP Route

Figure 1-7 Networking for IBGP route recursion



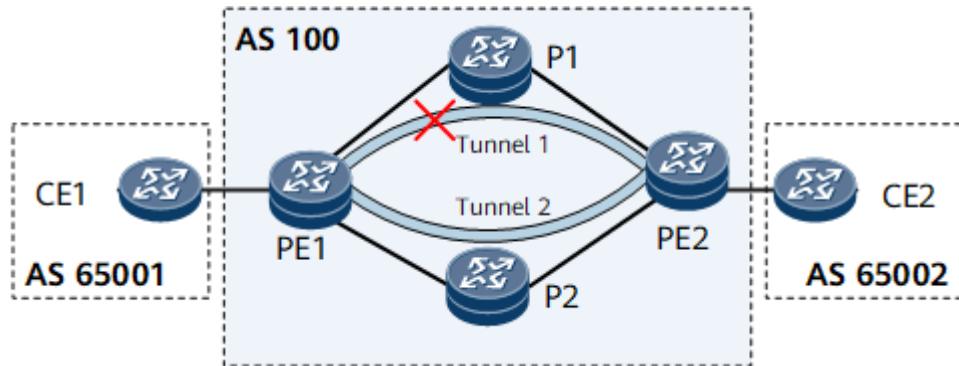
In [Figure 1-7](#), an IBGP peer relationship is established between Device A and Device D. The IBGP peer relationship is established between two loopback interfaces on the routers, but the next hop cannot be used to guide packet forwarding, because it is not directly reachable. Therefore, to refresh the forwarding table and guide packet forwarding, the system needs to search for the actual outbound interface and directly connected next hop based on the original IBGP next hop.

Device D receives 100,000 routes from Device A. These routes have the same original BGP next hop. After recursion, these routes eventually follow the same IGP path (A->B->D). If the IGP path (A->B->D) fails, these IBGP routes do not need to perform recursion separately, and the relevant forwarding entries do not need to be refreshed one by one. Note that only the shared next hop needs to perform recursion and be refreshed. Consequently, these IBGP routes converge to the path (A->C->D) on the forwarding plane. Therefore, the convergence time depends on only the number of next hops, not the number of prefixes.

If Device A and Device D establish a multi-hop EBGP peer relationship, the convergence procedure is the same as the preceding one. Indirect next hop also applies to the recursion of a multi-hop EBGP route.

VPN Routes Recursion to a Tunnel

Figure 1-8 Networking for VPN route recursion



In [Figure 1-8](#), a neighbor relationship is established between PE1 and PE2, and PE2 receives 100,000 VPN routes from PE1. These routes have the same original BGP next hop. After recursion, these VPN routes eventually follow the same public network tunnel (tunnel 1). If tunnel 1 fails, these routes do not need to perform recursion separately, and the relevant forwarding entries do not need to be refreshed one by one. Note that only the shared next hop needs to perform recursion, and the relevant forwarding entries need to be refreshed. Consequently, these VPN routes converge to tunnel 2 on the forwarding plane. In this manner, the convergence time depends on only the number of next hops, not the number of prefixes.

Default Route

Default routes are special routes. In most cases, they are configured by administrators. Default routes can also be generated by dynamic routing protocols, such as OSPF and IS-IS.

Default routes are used only when no matching routing entry is available for packet forwarding in the routing table. A default route in the routing table is the route to the network 0.0.0.0 (with mask 0.0.0.0). You can check whether the default route is configured using the **display ip routing-table** command.

If the destination address of a packet does not match any entry in the routing table, the packet is sent along a default route. If no default route exists and the destination address of the packet does not match any entry in the routing table, the packet is discarded. An Internet Control Message Protocol (ICMP) packet is then sent, informing the originating host that the destination host or network is unreachable.

Multi-Topology

Multi-Topology Overview

On a traditional IP network, only one unicast topology exists, and only one unicast forwarding table is available on the forwarding plane, which forces services transmitted from one router to the same destination address to share the same

next hop, and various end-to-end services, such as voice and data services, to share the same physical links. As a result, some links may become heavily congested whereas others remain relatively idle. To address this problem, configure multi-topology to divide a physical network into different logical topologies for different services.

By default, the base topology is created on the public network. The class-specific topology can be added or deleted in the public network address family view. Each topology contains its own routing table. The class-specific topology supports the addition, deletion, and import of protocol routes.

The base topology cannot be deleted.

Direct Routes Supporting Multi-Topology

Direct routes can be added to or deleted from the routing table of any topology. The same routes can also be added to multiple topologies, independent of each other.

Direct routes associated with interfaces are added to the base topology by default. Direct routes in the base topology are not deleted, and the base topology contains all direct routes.

Static Routes Supporting Multi-Topology

Static routes can be added to or deleted from the routing table of any topology. The routes with the same prefix, outbound interface, and next hop can also be added to multiple topologies, independent of each other.

Static routes, by default, are configured in the base topology. However, they can be configured in a specified class-specific topology and can be changed or deleted.

Static routes have no outbound interfaces, and therefore, need to perform recursion based on the next hop. In this case, you cannot specify the topology in which the next hop resides.

Public network static route recursion to a VPN next hop or VPN static route recursion to a public network next hop can be configured only in the base topology. When configuring static routes, you cannot specify the name of the topology in which the destination resides.

Association Between Direct Routes and a VRRP Group

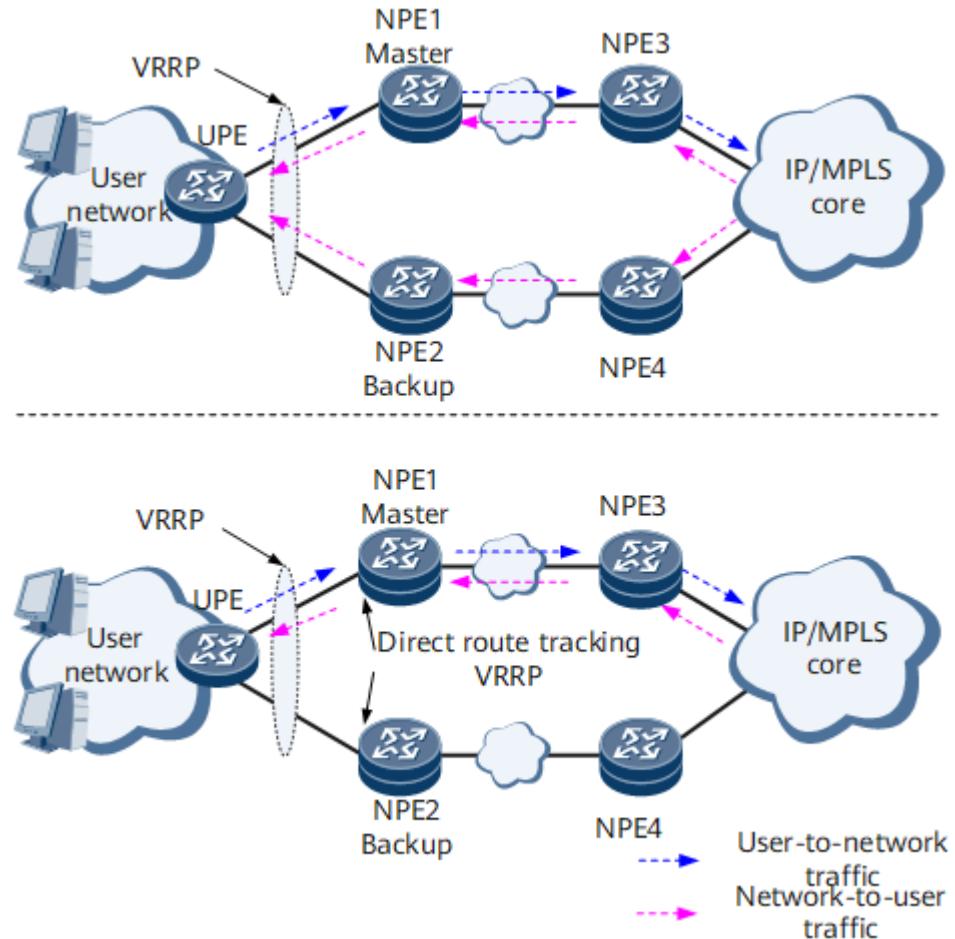
Background

A VRRP group is configured on Device1 and Device2 on the network shown in [Figure 1-9](#). Device1 is a master device, whereas Device2 is a backup device. The VRRP group serves as a gateway for users. User-to-network traffic travels through Device1. However, network-to-user traffic may travel through Device1, Device2, or both of them over a path determined by a dynamic routing protocol. Therefore, user-to-network traffic and network-to-user traffic may travel along different paths, which interrupts services if firewalls are attached to devices in the VRRP group, complicates traffic monitoring or statistics collection, and increases costs.

To address the preceding problems, the routing protocol is expected to select a route passing through the master device so that the user-to-network and network-

to-user traffic travels along the same path. Association between direct routes and a VRRP group can meet expectations by allowing the dynamic routing protocol to select a route based on the VRRP status.

Figure 1-9 Association between direct routes and a VRRP group



Related Concepts

VRRP is a widely used fault-tolerant protocol that groups multiple routing devices into a VRRP group, improving network reliability. A VRRP group consists of a master device and one or more backup devices. If the master device fails, the VRRP group switches services to a backup device to ensure communication continuity and reliability.

A device in a VRRP group operates in one of three states:

- **Master:** If a network is working correctly, the master device transmits all services.
- **Backup:** If the master device fails, the VRRP group selects a backup device as the new master device to take over traffic and ensure uninterrupted service transmissions.
- **Initialize:** A device in the Initialize state is waiting for an interface Startup message to switch its status to Master or Backup.

 NOTE

For details about VRRP, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description - Network Reliability - VRRP*.

Implementation

Association between direct routes and a VRRP group allows VRRP interfaces to adjust the costs of direct network segment routes based on the VRRP status. The direct route with the master device as the next hop has the lowest cost. A dynamic routing protocol imports the direct routes and selects the direct route with the lowest cost. For example, VRRP interfaces on Device1 and Device2 on the network shown in [Figure 1-9](#) are configured with association between direct routes and the VRRP group. The implementation is as follows:

- Device1 in the Master state sets the cost of its route to the directly connected virtual IP network segment to 0 (default value).
- Device2 in the Backup state increases the cost of its route to the directly connected virtual IP network segment.

A dynamic routing protocol selects the route with Device1 as the next hop because this route costs less than the other route. Therefore, both user-to-network traffic and network-to-user traffic travel through Device1.

Usage Scenario

When a data center is used, firewalls are attached to devices in a VRRP group to improve network security. Network-to-user traffic cannot pass through a firewall if it travels over a path different than the one used by user-to-network traffic.

When an IP radio access network (RAN) is configured, VRRP is configured to set the master/backup status of aggregation site gateways (ASGs) and radio service gateways (RSGs). Network-to-user and user-to-network traffic may pass through different paths, complicating network operation and management.

Association between direct routes and a VRRP group can address the preceding problems by ensuring the user-to-network and network-to-user traffic travels along the same path.

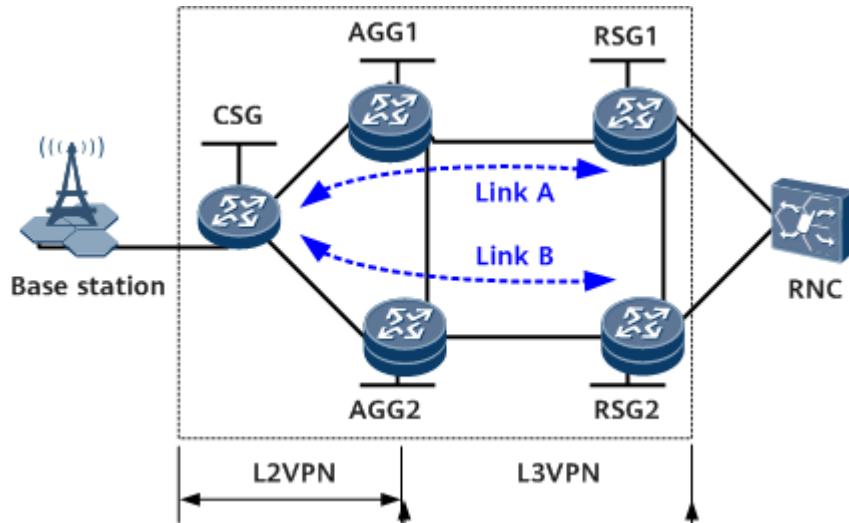
Direct Routes Responding to VE Interface Status Changes After a Delay

Background

On the network shown in [Figure 1-10](#), an L2VPN connection is set up between each AGG and the CSG, and BGP VPNv4 peer relationships are set up between the AGGs and RSGs. Layer 3 Virtual Ethernet (L3VE) interfaces are configured on AGGs and VPN instances are bound to the L3VE interfaces, so that the CSG can access the Layer 3 Virtual Private Network (L3VPN). The AGGs are configured to import direct routes into BGP and advertise these direct routes to their VPNv4 peers (RSGs). If AGG1 restarts or the CSG-AGG1 link fails, traffic switches over to Link B. After AGG1 or the CSG-AGG1 link recovers, the VE interface on AGG1 goes from down to up, and AGG1 immediately generates a direct route and advertises the route to the RSGs. The downstream traffic then switches back to Link A. However, AGG1 has not learned the MAC address of the base station yet. As a result, downstream traffic is lost.

To prevent this problem, configure RSGs not to preferentially select the routes advertised by AGG1 until AGG1 learns the MAC address of the base station. In this case, you can configure a delay for direct routes to respond to VE interface status changes.

Figure 1-10 Networking for direct routes responding to VE interface status changes after a delay



Implementation

After a delay is configured for direct routes to respond to VE interface status changes after a delay, the cost of the direct route is modified to the configured cost (greater than the default value 0) when the VE interface on AGG1 goes from down to up. After the configured delay expires, the cost of the direct route restores to the default value 0. Because BGP has imported the direct route and has advertised it to RSGs, the cost value determines whether RSGs preferentially select the direct route.

RSGs preferentially transmit traffic over Link B before AGG1 learns the MAC address of the base station, which prevents traffic loss.

Usage Scenario

Direct routes responding to VE interface status changes after a delay applies to IP radio access networks (RANs) on which an L2VPN accesses an L3VPN.

Association Between the Direct Route and PW Status

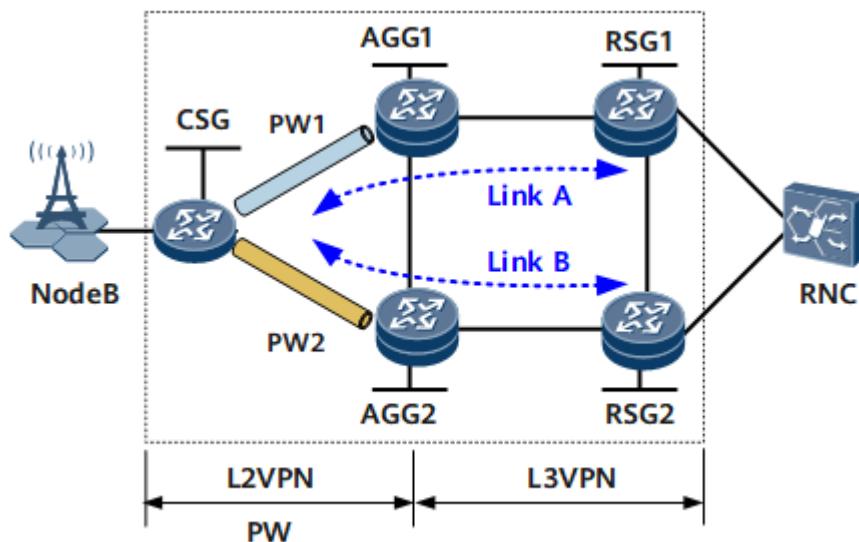
Background

In [Figure 1-11](#), PWs are set up between the AGGs and the CSG. BGP virtual private network version 4 (VPNv4) peer relationships are set up between the AGGs and RSGs. Layer 3 virtual Ethernet (L3VE) interfaces are configured on the AGGs, and VPN instances are bound to the L3VE interfaces so that the CSG can access the L3VPN. BGP is configured on the AGGs to import direct routes between the CSG and AGGs. The AGGs convert these direct routes to BGP VPNv4 routes before advertising them to the RSGs.

AGG1 functions as the master device in [Figure 1-11](#). In most cases, the RSGs select routes advertised by AGG1, and traffic travels along Link A. If AGG1 or the CSG-AGG1 link fails, traffic switches over to Link B. After AGG1 or the CSG-AGG1 link recovers, the L3VE interface on AGG1 goes from Down to Up, and AGG1 immediately generates a direct route destined for the CSG and advertises the route to the RSGs. Downstream traffic then switches over to Link A. However, PW1 is on standby. As a result, downstream traffic is lost.

To address this problem, associate the direct route and PW status. After the association is configured, the RSG preferentially selects the direct route only after PW1 becomes active.

Figure 1-11 Networking for the association between the direct route and PW status



Implementation

Configuring the association between the direct route and PW status allows a VE interface to adjust the cost value of the direct route based on PW status. The cost value determines whether the RSGs preferentially select the direct route because BGP has imported the direct route and has advertised it to RSGs. For example, if you associate the direct route and PW status on the network shown in [Figure 1-11](#), the implementation is as follows:

- When PW1 becomes active, the cost value of the direct route between the CSG and AGG1 restores to the default value 0. RSGs preferentially transmit traffic over Link A.
- When PW1 is on standby, the cost value of the direct route between the CSG and AGG1 is modified to a configured value (greater than 0). RSGs preferentially transmit traffic over Link B, which reduces traffic loss.

Usage Scenario

This feature applies to IP radio access networks (RANs) on which primary/secondary PWs are configured between the CSG and AGGs.

Vlink Direct Route Advertisement

Background

By default, ARP or IPv6 NDP Vlink direct routes are used only for data forwarding in the same VLAN. To control the routing table size and maintain routing table stability, these direct routes cannot be imported to dynamic routing protocols for advertisement. In some cases, the device needs to perform operations based on specific routes of VLAN users. For example, a device needs to apply a unique export policy for each VLAN user to divert traffic from a remote device. In this case, ARP or IPv6 NDP Vlink routes need to be imported to a dynamic routing protocol and advertised to the remote device.

Related Concepts

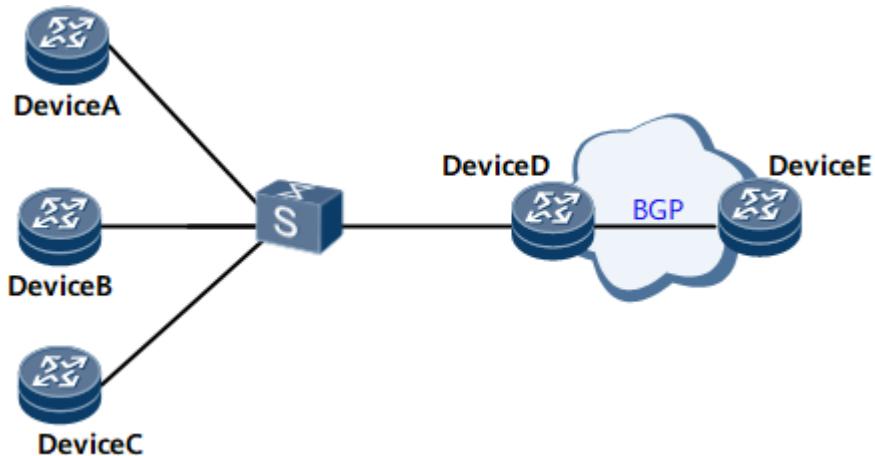
ARP Vlink direct routes: routes with VLAN users' physical interfaces that are learned using ARP. Such routes are used to forward IP packets in VLAN networking because packets cannot be forwarded through some logical interfaces. After the VLANIF interface, QinQ interface, or dot1q VLAN tag termination sub-interface learns an ARP entry of a remote end, an ARP Vlink direct route with a 32-bit mask is generated and displayed in the routing table. A common physical interface does not generate an ARP Vlink direct route with a 32-bit mask.

NDP Vlink direct routes: routing entries carrying IPv6 addresses of VLAN users' physical interfaces. These IPv6 addresses are learned and resolved using NDP.

Implementation

On the network shown in [Figure 1-12](#), three users (Device A, Device B, and Device C) are connected to the logical interface of Device D. Device E needs to communicate with Device B, not with Device A or Device C. In this scenario, Vlink direct route advertisement can be enabled on Device D. Then Device D obtains each physical interface of the three users and uses a route-policy to filter out network segment routes and the routes destined for Device A and Device C.

Figure 1-12 Networking for Vlink direct route advertisement



Usage Scenario

Vlink direct route advertisement is applicable to networks in which a device needs to add Vlink direct routes with physical interfaces of VLAN users to the routing table of a dynamic routing protocol before advertising the routes to remote ends.

Benefits

With Vlink direct route advertisement, a device can import Vlink direct routes into the routing table of a dynamic routing protocol (such as an IGP or BGP) and then use different export policies during route advertisement to implement precise route control.

Association Between the IPv4 Direct Routes and IPsec Instance Status

[Figure 1-13](#) shows an IP radio access network (IPRAN) scenario, and some services require high security. To meet such requirements, cell site gateways (CSGs) encrypt data of these services using IPsec. After the data flows to RSGs (IPsec gateways) through an IPsec tunnel, the RSGs decrypt the data. In most cases, carriers deploy master and backup RSGs and configure the same IP address for the IPsec tunnel interfaces of the master and backup RSGs to improve network reliability.

Without the association between IPv4 direct routes and IPsec instance status, IPv4 direct routes with the same prefix generated on the IPsec tunnel interfaces of the master and backup RSGs share the same default cost (0). As a result, after receiving these routes from the master and backup RSGs, CSGs cannot select an optimal one based on the cost.

Association between IPv4 direct routes and IPsec instance status can address this problem. After the association is configured:

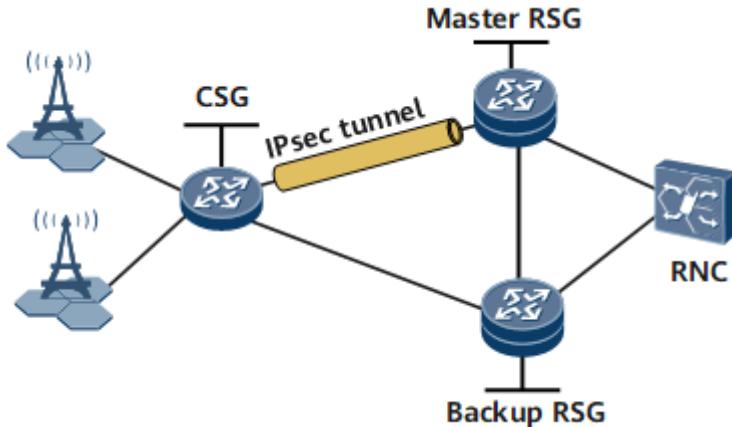
- If the IPsec instance status is master on an IPsec tunnel interface, the cost of the IPv4 direct routes generated on the interface is 0.
- If the IPsec instance status is backup on an IPsec tunnel interface or the system cannot detect the IPsec instance status, the cost of the IPv4 direct routes generated on the interface is the cost configured on the interface.

NOTE

In IPsec dual-device hot backup scenarios, if a tunnel interface borrows the IP address of another interface, the cost of direct routes on the tunnel interface cannot be associated with the IPsec instance status.

After receiving the IPv4 direct routes with the same prefix from the master and backup RSGs, CSGs can select an optimal one based on the cost. Therefore, the CSGs can transmit data encrypted using IPsec to the correct RSG.

Figure 1-13 Networking for the association between IPv4 direct routes and IPsec instance status

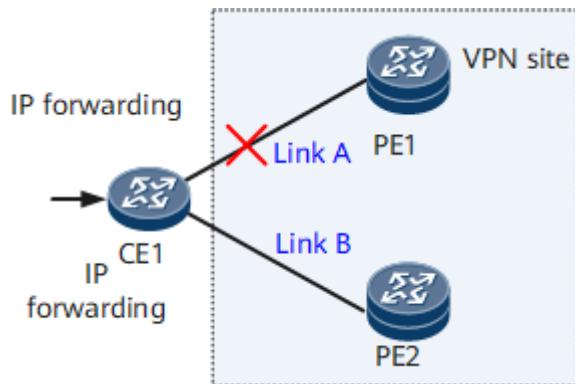


1.1.1.1.3 Application Scenarios for IP Routing

Typical Application of IP FRR

In **Figure 1-14**, CE1 is dual-homed to PE1 and PE2. CE1 is configured with two outbound interfaces and two next hops. Link B functions as the backup of link A. If link A fails, traffic can be rapidly switched to link B.

Figure 1-14 Configuring IP FRR



Data Center Applications of Association Between Direct Routes and a VRRP Group

Service Overview

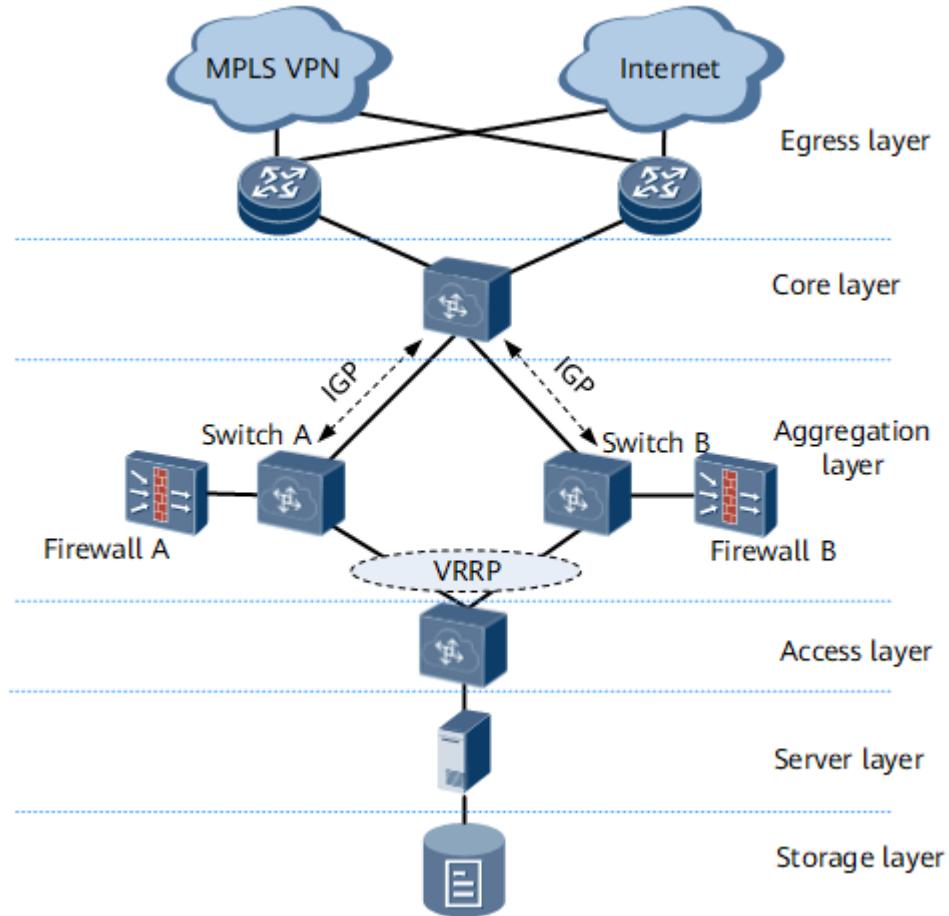
A data center, used for service access and transmission, consists of many servers, disk arrays, security devices, and network devices that store and process a great number of services and applications. Firewalls are used to improve data security, and VRRP groups are configured to improve communication reliability. VRRP may cause user-to-network traffic and network-to-user traffic to travel along different paths, and as a result, the firewall may discard the network-to-user traffic because

of path inconsistency. To address this problem, association between direct routes and a VRRP group must be configured.

Networking Description

Figure 1-15 shows a data center network. A server functions as a core service module in the data center. A VRRP group protects data exchanged between the server and core devices, improving service security. Firewalls are attached to devices in the VRRP group to improve network security.

Figure 1-15 Data center network



Feature Deployment

The master device transmits server traffic to a core device. When the core device attempts to send traffic to the server, the traffic can only pass through a firewall attached to the master device. On the network shown in **Figure 1-15**, the server sends data destined for the core device through the master device, and the core device sends data destined for the server along a path that an Interior Gateway Protocol (IGP) selects. The association between the direct routes and a VRRP group can be configured on switch A and switch B so that the IGP selects a route based on VRRP status. The IGP forwards core-device-to-server traffic over the same path as the one over which server-to-core-device traffic is transmitted, which prevents the firewall from discarding traffic.

Application of Association Between Direct Routes and a VRRP Group on an IP RAN

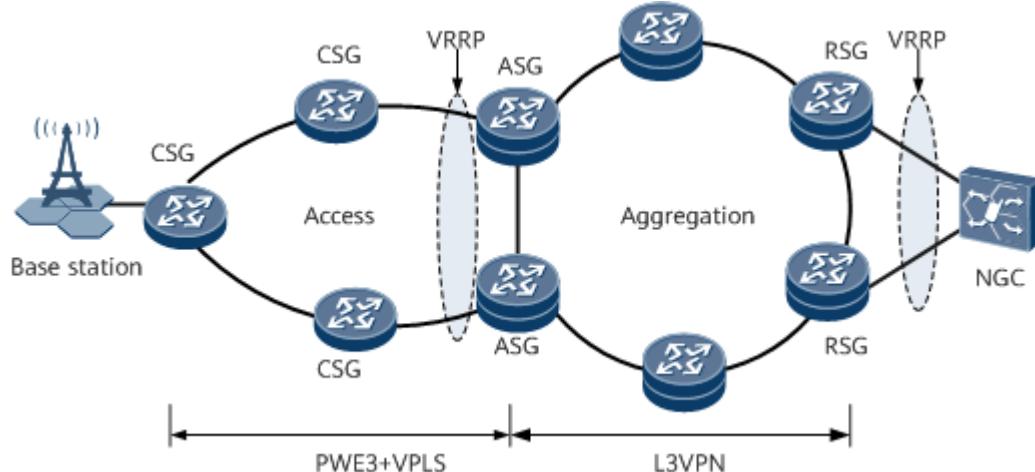
Service Overview

In an IP RAN scenario, base stations and NGCs do not support dynamic routing configuration. Therefore, you need to configure static routes with the addresses of ASGs and RSGs specified for communication with the aggregation network. VRRP is configured to provide ASG and RSG redundancy, improving device reliability and ensuring non-stop transmission of services, such as voice, video, and cloud application services over mobile transport networks.

Networking Description

Figure 1 shows VRRP-based gateway protection applications on an IP RAN. A base station is dual-homed to VRRP-enabled ASGs to communicate with the aggregation network. The traffic from the base station to the NGC passes through the master ASG in the VRRP group, whereas the traffic from the NGC to the base station is transmitted over a path selected by a dynamic routing protocol after leaving the aggregation network. Similarly, the RSGs connecting the NGC and the aggregation network also use VRRP to implement redundancy. The upstream and downstream traffic paths on the NGC side also have the same problem.

Figure 1-16 VRRP application in the IP RAN gateway protection solution



Feature Deployment

According to the networking description, the upstream and downstream traffic paths on both sides of the aggregation network may be inconsistent. Path inconsistency causes operation and management difficulties. For example, upstream traffic enters the aggregation network through the master ASG, whereas downstream traffic flows out of the aggregation network through the backup ASG. Path inconsistency complicates traffic monitoring or statistics collection and increases the cost. In addition, the VRRP technology is used to implement device-level redundancy backup. When the master device is running properly, the backup device does not carry any traffic. Therefore, the fact that the backup ASG carries downstream traffic does not comply with the original intention of the redundancy backup design. Association between direct routes and a VRRP group can be configured to ensure path consistency.

On the base station side, you can associate direct routes with a VRRP group on ASG interfaces configured with service VRRP, and import direct routes into a dynamic routing protocol on the aggregation network. The VRRP status affects route selection of dynamic routing protocols, ensuring that the route that passes through the master ASG is always preferentially selected. In this manner, upstream and downstream traffic is transmitted along the same path. Implementation on the NGC side is similar to that on the base station side.

1.1.1.1.4 Appendix List of Port Numbers of Common Protocols

Table 1-6 Port numbers of routing protocols

Routing Protocol	UDP Port Number	TCP Port Number
RIP	520	-
RIPv2	520	-
RIPng	521	-
BGP	-	179
OSPF	-	-
IS-IS	-	-

Note that "-" indicates that the related transport layer protocol is not used.

Table 1-7 Port numbers of application layer protocols

Application Layer Protocol	UDP Port Number	TCP Port Number
DHCP	67/68	-
DNS	53	53
FTP	-	20/21
HTTP	-	80
IMAP	-	993
NetBIOS	137/138	137/139
POP3	-	995
SMB	445	445
SMTP	25	25
SNMP	161	-
TELNET	-	23
TFTP	69	-

Application Layer Protocol	UDP Port Number	TCP Port Number
Note that "-" indicates that the related transport layer protocol is not used.		

1.1.1.5 Terminology for IP Routing

Terms

Term	Description
ARP Vlink direct routes	IP packets are forwarded through a specified physical interface. IP packets cannot be forwarded through a VLANIF interface, because a VLANIF interface is a logical interface with several physical interfaces as its member interfaces. If an IPv4 packet reaches a VLANIF interface, the device obtains information about the physical interface using ARP and generates the relevant routing entry. The route recorded in the routing entry is called an ARP Vlink direct route.
FRR	FRR is applicable to services that are very sensitive to packet loss and delay. When a fault is detected at the lower layer, the lower layer informs the upper layer routing system of the fault. Then, the routing system forwards packets through a backup link. In this manner, the impact of the link fault on services is minimized.
NDP Vlink direct routes	IP packets are forwarded through a specified physical interface. IP packets cannot be forwarded through a VLANIF interface, because a VLANIF interface is a logical interface with several physical interfaces as its member interfaces. If an IPv6 packet reaches a VLANIF interface, the device obtains information about the physical interface using the neighbor discovery protocol (NDP) and generates the relevant routing entry. The route recorded in the routing entry is called an NDP Vlink direct route.
UNR	When a user goes online through a Layer 2 device, such as a switch, but there is no available Layer 3 interface and the user is assigned an IP address, no dynamic routing protocol can be used. To enable devices to use IP routes to forward the traffic of this user, use the Huawei User Network Route (UNR) technology to assign a route to forward the traffic of the user.

Abbreviations

Abbreviation	Full Name
ARP	Address Resolution Protocol

Abbreviation	Full Name
BGP	Border Gateway Protocol
CE	Customer Edge
FIB	Forwarding Information Base
IBGP	Internal Border Gateway Protocol
IGP	Internal Gateway Protocol
IS-IS	Intermediate System-Intermediate System
NDP	Neighbor Discover Protocol
OSPF	Open Shortest Path First
PE	Provider Edge
RIP	Routing Information Protocol
RM	Route Management
Vlink	Virtual Link
VoIP	Voice Over IP
VPN	Virtual Private Network
VRP	Versatile Routing Platform

1.1.1.2 Basic IP Routing Configuration

IP routing is the basic concept on data communication networks.

1.1.1.2.1 Overview of Basic IP Routing

To forward data, routers must be capable of establishing and refreshing routing tables and forwarding datagrams based on routing tables.

Definition

As a basic concept on data communication networks, routing is the process of packet relaying or forwarding, and the process provides route information for packet forwarding.

Purpose

During data forwarding, routers, routing tables, and routing protocols are indispensable. Routing protocols are used to discover routes and contribute to the generation of routing tables. Routing tables store the routes discovered by various routing protocols, and routers select routes and implement data forwarding.

1.1.1.2.2 Configuration Precautions for Basic IP Routing

Feature Requirements

Table 1-8 Feature requirements

Feature Requirements	Series	Models
<p>Backup routes cannot be configured for mutual access between public and private networks. If backup routes are configured for public and private network routes, only the forwarding of public and private network routes is ensured. The forwarding of backup routes and the switchover between public and private network routes and their backup routes cannot be ensured.</p> <p>Plan services properly.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

1.1.1.2.3 Configuring the Router ID

The router ID uniquely identifies a device in an AS.

Usage Scenario

A router ID is a 32-bit IP address that uniquely identifies a router in an Autonomous System (AS). A router ID can be generated as follows:

- Manually configured
- Configured by the protocol
- Automatically selected

 NOTE

If the IP address of a physical interface is configured as the router ID and then the IP address changes, route flapping may occur. Therefore, configuring the IP address of a loopback interface as the router ID is recommended.

Pre-configuration Tasks

None

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **router id router-id**

The Router ID is specified.

Step 3 Run commit

The configuration is committed.

----End

Checking the Configurations

After configuring a router ID, check it.

- Run the **display router id** command to check the router ID.

1.1.1.2.4 Configuring IPv4 Multi-Topology

Multi-topology on an IPv4 network enables devices to allocate network resources more effectively.

Usage Scenario

On traditional IP networks, only one unicast forwarding table is available on the forwarding plane because only one unicast topology exists, which forces services transmitted from one router to the same destination address to share the same next hop, and various end-to-end services, such as voice and data services, to share the same physical links. As a result, some links may be heavily congested while others remain relatively idle. Deploying MT (different logical topologies) for different services on a physical network can solve the problem.

Pre-configuration Tasks

Before configuring IPv4 Multi-Topology, complete the following task:

- Configure parameters of the link layer protocol and IPv4 addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ip topology *topology-name*

A topology is created.

Step 3 Run interface *interface-type interface-name*

The view of the specified interface is displayed.

Step 4 Run ip topology *topology-name* enable

The topology is bound to the interface.

Step 5 Run commit

The configuration is committed.

----End

Checking the Configurations

Run the **display ip topology** command to check the previous configuration.

1.1.1.2.5 Configuring IPv6 Multi-Topology

Multi-topology on an IPv6 network enables devices to allocate network resources more effectively.

Usage Scenario

On traditional IPv6 networks, only one unicast forwarding table is available on the forwarding plane because only one unicast topology exists, which forces services transmitted from one router to the same destination address to share the same next hop, and various end-to-end services, such as voice and data services, to share the same physical links. As a result, some links may be heavily congested while others remain relatively idle. Deploying MT (different logical topologies) for different services on a physical network can solve the problem.

Pre-configuration Tasks

Before configuring IPv6 multi-topology, configure parameters of the link layer protocol and IPv6 addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ipv6 topology topology-name**

An IPv6 topology is created.

Step 3 Run **interface interface-type interface-name**

The view of the specified interface is displayed.

Step 4 Run **ipv6 topology topology-name enable**

The topology is bound to the interface.

Step 5 Run **commit**

The configuration is committed.

----End

Checking the Configurations

Run the **display ipv6 topology** command to check the previous configuration.

1.1.1.2.6 Configuring IPv4 FRR

IPv4 FRR is applicable to IPv4 services that are sensitive to the packet loss and delay.

Usage Scenario

Public network IPv4 FRR is applicable to services that are sensitive to packet loss or delay on the IPv4 public network.

After FRR is configured, if a fault is detected at the lower layer, the fault is reported to the upper-layer routing system. Then, packets are forwarded through a backup link, which minimizes the impact of link faults on ongoing services.

NOTICE

IPv4 FRR enables routes of different protocols to back up each other, which may result in loops.

With IP FRR, traffic is switched to a backup link if the primary link fails and switched back when the primary link recovers. If the inbound and outbound interfaces reside on different boards, packet loss may occur during the switchback. The packet loss duration varies with the service volume and CPU usage. To prevent the packet loss, perform any of the following operations:

- For IS-IS routes, run the **timer spf** command in the IS-IS view.
- For OSPF routes, run the **spf-schedule-interval** command in the OSPF view.
- For BGP routes, run the **route-select delay** command in the BGP view.

Pre-configuration Tasks

Before configuring IPv4 FRR, complete the following task:

- Configure parameters of the link layer protocol and IPv4 addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.
- Configure routes of different routing protocols but destined for the same destination address.

Enabling IPv4 FRR

Before configuring IPv4 FRR, you need to enable IPv4 FRR globally.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip frr**

IPv4 FRR is enabled.

NOTE

If IPv4 FRR is configured both in the system view and a routing protocol view, the IPv4 FRR that is configured in the routing protocol view preferentially takes effect.

Step 3 Run **commit**

The configuration is committed.

----End

(Optional) Enabling IPv4 FRR Poison Reverse

To configure IPv4 FRR on an IP ring network, you need to enable IPv4 FRR poison reverse to prevent instantaneous traffic storms during route convergence.

Context

In [Figure 1-17](#), IPv4 FRR is deployed on the IP ring network, and traffic enters the ring network from Device D and leaves the ring network from Device A. The primary and backup next hops of Device D are Device C and Device E, respectively. The primary and backup next hops of Device C are Device B and Device D, respectively. If the link between Device B and Device C fails and the traffic reaches Device C, Device C determines that the primary link fails and forwards the traffic to Device D. However, before the route convergence is complete, Device D cannot detect the link failure between Device B and Device C, and continues forwarding the traffic to Device C. As a result, an instantaneous traffic storm is generated between Device C and Device D. After Device D detects the link failure between Device B and Device C through route convergence, Device D forwards the traffic to its backup next hop (Device E). To prevent instantaneous traffic storms during route convergence and enable Device D to detect the link failure quickly, you can enable the IPv4 FRR poison reverse function. After the function is enabled, Device D searches its FIB based on the destination IP address to obtain the next hop and outbound interface+VLAN information. If the outbound interface is the primary interface and is up, Device D checks whether the outbound interface+VLAN is the same as the inbound interface+VLAN. If they are the same, Device D forwards the traffic through the backup interface. If the link between Device B and Device C fails, the traffic forwarded from Device D to Device C goes back to Device D. Device D does not send the traffic back to Device C; instead, it forwards the traffic to its backup next hop (Device E). Finally, the traffic leaves the ring network from Device A, as shown in [Figure 1-18](#).

Figure 1-17 IP ring network

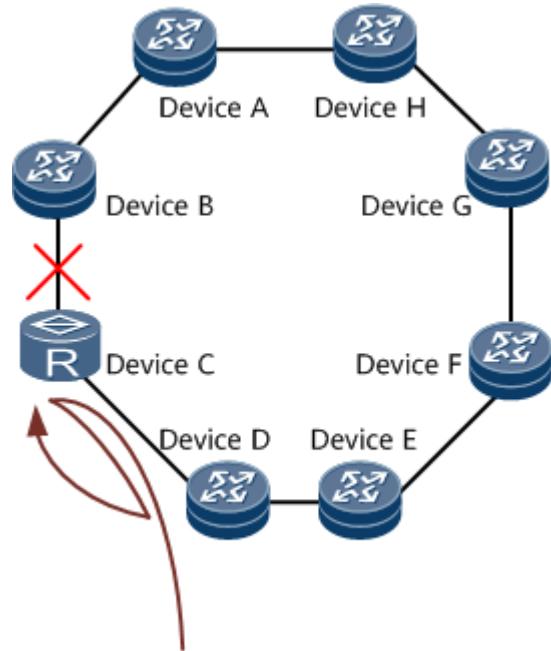
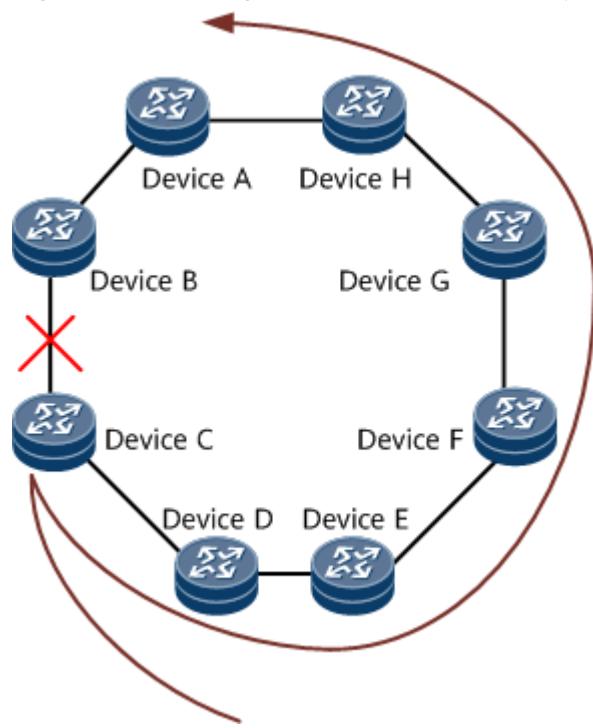


Figure 1-18 IP ring network with IPv4 FRR poison reverse enabled



Procedure

Step 1 Run `interface interface-type interface-number`

The interface view or sub-interface view is displayed.

Step 2 Run `poison-reverse enable`

IPv4 FRR poison reverse is enabled.

This command is used to prevent instantaneous traffic storms during route convergence in a scenario where IPv4 FRR is configured on an IP ring network.

Poison reverse does not support load balancing.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

After configuring IPv4 FRR, you can view information about the backup outbound interfaces and backup next hops in the routing table.

Prerequisites

IPv4 FRR has been configured.

Procedure

- Run the **display ip routing-table verbose** command to check backup outbound interface and backup next hop information in the routing table.
- Run the **display ip routing-table ip-address [mask | mask-length] [longer-match] verbose** command to check backup outbound interface and backup next hop information in the routing table.
- Run the **display ip routing-table ip-address1 { mask1 | mask-length1 } ip-address2 { mask2 | mask-length2 } verbose** command to check backup outbound interface and backup next hop information in the routing table.

----End

1.1.1.2.7 Configuring IPv6 FRR

IPv6 FRR is applicable to services that are sensitive to the delay and packet loss on an IPv6 network.

Usage Scenario

Public network IPv6 FRR is applicable to services that are sensitive to the delay and packet loss on an IP public network.

After IPv6 FRR is configured, if a fault is detected at the lower layer, the fault is reported to the upper-layer routing system. Then, packets are forwarded through a backup link, which minimizes the impact of link faults on ongoing services.

NOTICE

IPv6 FRR enables routes of different protocols to back up each other, which may result in loops.

Pre-configuration Tasks

Before configuring IPv6 FRR, complete the following tasks:

- Configure parameters of the link layer protocol and IPv6 addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.
- Configure routes of different routing protocols but destined for the same destination address.

Enabling IPv6 FRR

Before configuring IPv6 FRR, you need to enable IPv6 FRR globally.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ipv6 frr

IPv6 FRR is enabled.



If IPv6 FRR is configured both in the system view and a routing protocol view, the IPv6 FRR that is configured in the routing protocol view preferentially takes effect.

Step 3 Run commit

The configuration is committed.

----End

(Optional) Enabling IPv6 FRR Poison Reverse

To configure IPv6 FRR on an IP ring network, you need to enable IPv6 FRR poison reverse to prevent instantaneous traffic storms during route convergence.

Context

In [Figure 1-19](#), IPv6 FRR is deployed on the IP ring network, and traffic enters the ring network from Device D and leaves the ring network from Device A. The primary and backup next hops of Device D are Device C and Device E, respectively. The primary and backup next hops of Device C are Device B and Device D, respectively. If the link between Device B and Device C fails and the traffic reaches Device C, Device C determines that the primary link fails and forwards the traffic to Device D. However, before the route convergence is complete, Device D cannot detect the link failure between Device B and Device C, and continues forwarding the traffic to Device C. As a result, an instantaneous traffic storm is generated between Device C and Device D. After Device D detects the link failure between Device B and Device C through route convergence, Device D forwards the traffic to its backup next hop (Device E). To prevent instantaneous traffic storms during route convergence and enable Device D to detect the link failure quickly, you can enable the IPv6 FRR poison reverse function. After the function is enabled, Device D searches its FIB based on the destination IP address to obtain the next hop and outbound interface+VLAN information. If the outbound interface is the primary

interface and is up, Device D checks whether the outbound interface+VLAN is the same as the inbound interface+VLAN. If they are the same, Device D forwards the traffic through the backup interface. If the link between Device B and Device C fails, the traffic forwarded from Device D to Device C goes back to Device D. Device D does not send the traffic back to Device C; instead, it forwards the traffic to its backup next hop (Device E). Finally, the traffic leaves the ring network from Device A, as shown in [Figure 1-20](#).

Figure 1-19 IP ring network

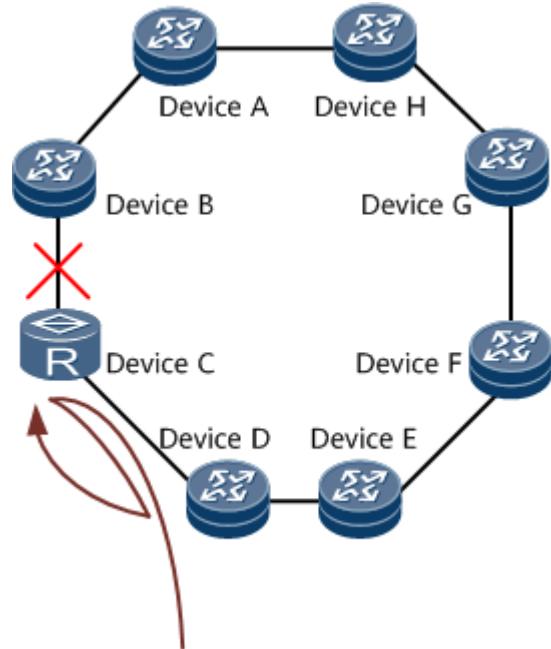
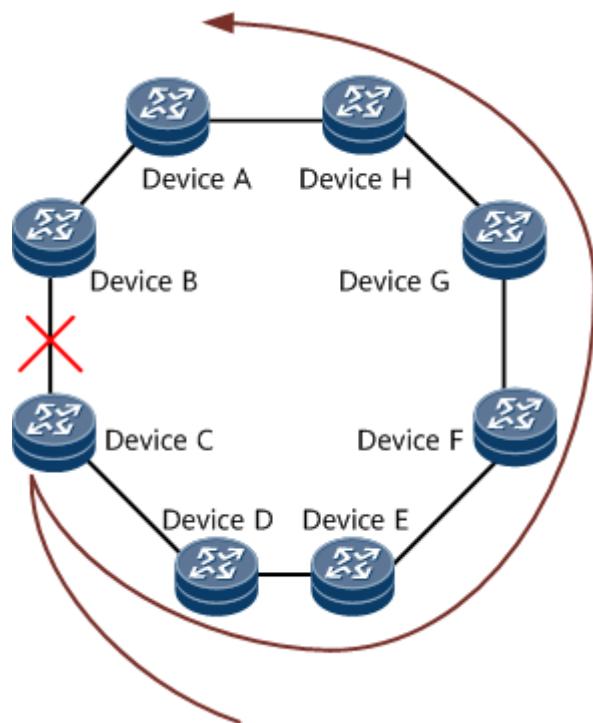


Figure 1-20 IP ring network with IPv6 FRR poison reverse enabled



Procedure

Step 1 Run **interface interface-type interface-number**

The interface view or sub-interface view is displayed.

Step 2 Run **poison-reverse enable**

IPv6 FRR poison reverse is enabled.

This command is used to prevent instantaneous traffic storms during route convergence in a scenario where IPv6 FRR is configured on an IP ring network.

Poison reverse does not support load balancing.

Step 3 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring IPv6 FRR, you can view information about the backup outbound interfaces and backup next hops in the routing table.

Prerequisites

IPv6 FRR has been configured.

Procedure

- Run the **display ipv6 routing-table verbose** command to check backup outbound interface and backup next hop information in the routing table.
- Run the **display ipv6 routing-table ip-address [mask | mask-length] [longer-match] verbose** command to check backup outbound interface and backup next hop information in the routing table.
- Run the **display ipv6 routing-table ip-address1 { mask1 | mask-length1 } ip-address2 { mask2 | mask-length2 } verbose** command to check backup outbound interface and backup next hop information in the routing table.

----End

1.1.1.2.8 Configuring Route Recursion to the Default Route

When the next hop of a route is not directly reachable, you can configure route recursion to the default route.

Usage Scenario

The next hops of routes may not be directly reachable. In this case, recursion is required so that such routes can be used for traffic forwarding. To allow routes to recurse to the default route, run the **{ ip | ipv6 } route recursive-lookup default-route** command. By default, routes cannot recurse to the default route.

Pre-configuration Tasks

None

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ip route recursive-lookup default-route protocol { static | msr } or ipv6 route recursive-lookup default-route protocol { static | msr }

Route recursion to the default route is enabled.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display current-configuration** command in the system view to verify it. The command output shows that route recursion to the default route is configured.

1.1.1.2.9 Configuring the Device to Advertise Host Routes of Directly Connected Interfaces

You can configure a device to advertise the host routes of directly connected interfaces after the routes are imported by routing protocols.

Usage Scenario

By default, after a routing protocol imports host routes of directly connected interfaces, the routing protocol stores them in the routing table but does not advertise them. You can configure the device to advertise host routes of directly connected interfaces as required.

Pre-configuration Tasks

None

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ip direct-routing-table local-host-route advertise enable or ipv6 direct-routing-table local-host-route advertise enable

The device is configured to advertise host routes of directly connected interfaces.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

After configuring the device to advertise host routes of directly connected interfaces, run the **display current-configuration** command in the system view. The command output shows that this function has been enabled.

1.1.1.2.10 Configuring Association Between Direct Routes and a VRRP Group

Association between direct routes and a Virtual Router Redundancy Protocol (VRRP) group ensures that both network-to-user traffic and user-to-network traffic that traverse the VRRP group travel along the same path, which facilitates network management and improves reliability.

Usage Scenario

On a live network, a VRRP group serves as a gateway for users (including common users and base stations) to access the network. User-to-network traffic traverses the master device in the VRRP group, while network-to-user traffic travels along a path that a dynamic routing protocol selects. Therefore, user-to-network traffic and network-to-user traffic may travel along different paths, which interrupts services if firewalls are attached to devices in the VRRP group, complicates traffic monitoring or statistics collection, and increases costs.

To address the preceding problems, the routing protocol is expected to select a route passing through the master device so that the user-to-network and network-to-user traffic travels along the same path. Association between direct routes and a VRRP group can meet expectations by allowing the dynamic routing protocol to select a route based on the VRRP status.

Pre-configuration Tasks

Before associating direct routes with a VRRP group, complete the following tasks:

- Configure VRRP basic functions and create a VRRP group.
- Configure a dynamic routing protocol to ensure reachable routes among nodes.

NOTE

With association between direct routes and a VRRP group, VRRP and an IGP cannot be configured on the same interface. If VRRP and an IGP are configured on the same interface, the IGP cannot inherit the route costs from the imported direct routes. As a result, the path selected by the IGP for network-to-user traffic may be different from the one on which user-to-network traffic is transmitted.

Associating Direct Routes with a VRRP Group

Associating direct routes with a VRRP group on a VRRP-enabled interface or a Loopback interface allows the interface to modify the cost of each direct route to the virtual IP network segment based on the VRRP status.

Context

If the master device in a VRRP group fails, a master/backup switchover is performed. After the master device recovers, the direct routes of the master device immediately become available. The VRRP group performs a VRRP switchback, which takes some time. To ensure that user-to-network traffic and network-to-user traffic travel along the same path in this period, associate direct routes with the VRRP group on both the master and backup devices. In this manner, the VRRP status affects the cost of the direct route to the virtual IP network segment of the VRRP group.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-name**

The interface view is displayed.

VRRP must have been configured on this interface.

Step 3 Configure the association between direct routes with the VRRP group status.

- Configure the association between IPv4 direct routes with the VRRP4 group status.
 - On non-loopback interfaces:
Run the **direct-route track vrrp vrid virtual-router-id degrade-cost cost-value** command.
 - On loopback interfaces:
Run the **direct-route track vrrp interface interface-type interface-number vrid virtual-router-id degrade-cost cost-value** command.
- To configure association between IPv6 direct routes with the VRRP6 group status, run the **direct-route ipv6 track vrrp6 vrid virtual-router-id degrade-cost cost-value** command.

After the command is run, the cost of direct routes is adjusted based on the VRRP group status, with details as follows:

- If the VRRP group status is Master, 0 (the highest priority) is used as the cost of the direct routes.
- If the VRRP group status is Backup or Initialize, the *cost-value* (greater than 0) specified in the command is used as the cost of the direct routes.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring a Dynamic Routing Protocol to Import Direct Routes

After you associate direct routes with a Virtual Router Redundancy Protocol (VRRP) group, configure a dynamic routing protocol to import the direct routes so that the path selection by the dynamic routing protocol can be controlled.

Context

After direct routes are associated with a VRRP group, VRRP-enabled devices modify the cost of each direct route to the virtual IP network segment based on the VRRP status. To control the path by a dynamic routing protocol, enable the dynamic routing protocol to import direct routes and inherit the costs from the imported direct routes.

Currently, dynamic routing protocols include the Interior Gateway Protocol (IGP) and Border Gateway Protocol (BGP). If an IGP is configured to import direct routes, routing information protocol (RIP) cannot inherit costs from the imported direct routes. This section describes how to configure Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS), and BGP to import direct routes.

Procedure

- Configure OSPF to import direct routes.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **ospf [process-id]**

The OSPF process view is displayed.
 - c. Run **import-route direct**

OSPF is configured to import direct routes.
 - d. Run **default cost inherit-metric**

OSPF is enabled to inherit the costs from the imported direct routes.

NOTE

- The **default (OSPF)** command has a lower priority than the **apply cost** command. If the **apply cost** command is run, it overrides the **default (OSPF)** command. Therefore, before running the **default (OSPF)** command, ensure that the **apply cost** command is not run.
- Do not run the **default cost cost-value** command after you run the **default cost inherit-metric** command. If the **default cost cost-value** command is run after the **default cost inherit-metric** command, the **default cost cost-value** command overrides the **default cost inherit-metric** command.

- e. Run **commit**

The configuration is committed.
- Configure IS-IS to import direct routes.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **isis [process-id]**

The IS-IS process view is displayed.
 - c. Run **import-route direct inherit-cost**

IS-IS is configured to import direct routes and to inherit the costs from the imported direct routes.

- d. Run **commit**
The configuration is committed.
- Configure BGP to import direct routes.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP process view is displayed.
 - c. Run **import-route direct**
BGP is configured to import direct routes.
BGP automatically inherits the costs from the imported direct routes. The inherited costs are shown as the multi-exit discrimination (MED) values.
 - d. Run **commit**
The configuration is committed.

----End

Verifying the Configuration of Association Between Direct Routes and a VRRP Group

After you associate direct routes with a Virtual Router Redundancy Protocol (VRRP) group, you can view information about the VRRP group and information in the routing table of the previous-hop device on the network side.

Prerequisites

Direct routes have been associated with a VRRP group.

Procedure

- Run the **display ip routing-table** command on the backup device in the VRRP group to check the modified direct route costs in the IP routing table.
- Run the **display ip routing-table ip-address** command on the VRRP group's previous-hop device on the network side to check whether network-to-user traffic flows through the master device in the VRRP group.

----End

1.1.1.2.11 Configuring an IPv4 Direct Route to Respond to L3VE Interface Status Changes After a Delay

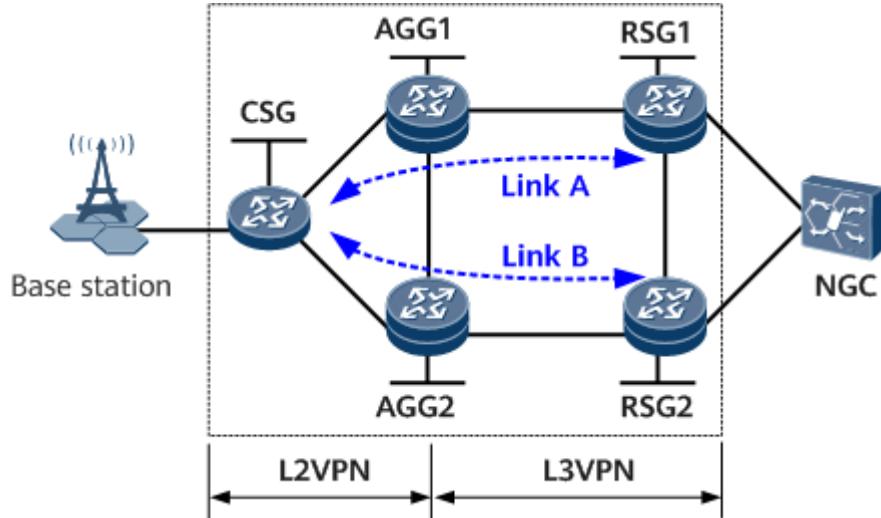
This section describes how to configure an IPv4 direct route to respond to Layer 3 virtual Ethernet (L3VE) interface status changes after a delay on an IP radio access network (RAN). During traffic switchback after the master AGG recovers, this configuration can reduce traffic loss and improve network reliability.

Usage Scenario

On the network shown in [Figure 1-21](#), a Layer 2 virtual private network (VPN) connection is set up between each AGG and the CSG through L2VE interfaces, and

Border Gateway Protocol (BGP) VPNv4 peer relationships are set up between the AGGs and RSGs on an L3VPN. L3VE interfaces are configured on the AGGs, and VPN instances are bound to the L3VE interfaces so that the CSG can access the L3VPN. BGP is configured on the AGGs to import IPv4 direct routes between the CSG and AGGs. The AGGs convert these IPv4 direct routes to BGP VPNv4 routes before advertising them to the RSGs.

Figure 1-21 Networking for the IPv4 direct route responding to L3VE interface status changes after a delay



AGG1 functions as the master device. In most cases, the RSGs select routes advertised by AGG1, and traffic travels along Link A. If AGG1 or the CSG-AGG1 link fails, traffic switches over to Link B. After AGG1 or the CSG-AGG1 link recovers, the L3VE interface on AGG1 goes from down to up, and AGG1 immediately generates an IPv4 direct route and advertises the route to the corresponding RSG. Downstream traffic then switches back to Link A. However, AGG1 has not learned the MAC address of the base station yet. As a result, downstream traffic is lost.

After a delay is configured for IPv4 direct routes to respond to L3VE interface status changes, the cost of the IPv4 direct route increases when the L3VE interface on AGG1 goes from down to up. In this case, RSGs do not preferentially select the route advertised by AGG1. Therefore, downstream traffic still travels along Link B. After the specified delay elapses, AGG1 learns the MAC address of the base station, and the cost of the IPv4 direct route restores to the default value 0. In this case, RSGs preferentially select the IPv4 direct route advertised by AGG1, preventing downstream traffic loss during switchback.

Pre-configuration Tasks

Before you configure an IPv4 direct route to respond to L3VE interface status changes after a delay, configure link-layer protocol parameters and IP addresses for interfaces and ensure that the link layer protocol of each interface is Up.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **interface virtual-ethernet interface-number**

A VE interface is created, and the VE interface view is displayed.

Step 3 Run **ve-group ve-group-id l3-access**

The VE interface is configured as an L3VE interface for MPLS L3VPN access and bound to a VE-group.



The L2VPN can access the L3VPN only if the L2VE and L3VE interfaces are bound to the same VE-group and reside on the same board.

Step 4 Run **quit**

Return to the system view.

Step 5 Run **interface virtual-ethernet interface-number.subinterface-number**

A VE sub-interface is created, and the VE sub-interface view is displayed.

Step 6 Run **direct-route degrade-delay delay-time degrade-cost cost**

The IPv4 direct route is configured to respond to L3VE interface status changes after a delay.

After you run the **direct-route degrade-delay** command on an L3VE interface, and the L3VE interface goes from Down to Up, the cost of the L3VE interface's IPv4 direct route to the CSG is modified to the configured cost. After the configured *delay-time* expires, the IPv4 direct route cost is restored to the default value 0.



The **direct-route track pw-state**, **direct-route degrade-delay**, and **direct-route track vrrp** commands cannot all be configured on one L3VE interface. If you run the three commands on one L3VE interface, the latest configuration overrides the previous ones.

Step 7 Run **commit**

The configuration is committed.

----End

Checking the Configurations

If an AGG's L3VE interface goes from Down to Up, run the **display ip routing-table vpn-instance vpn-instance-name [ip-address] [verbose]** command on the AGG. The command output shows that the cost of the L3VE interface's IPv4 direct route to the CSG is the configured cost. After the configured *delay-time* expires, run the **display ip routing-table vpn-instance vpn-instance-name [ip-address] [verbose]** command on the AGG. The command output shows that the cost of the IPv4 direct route to the L3VE interface is the default value 0.

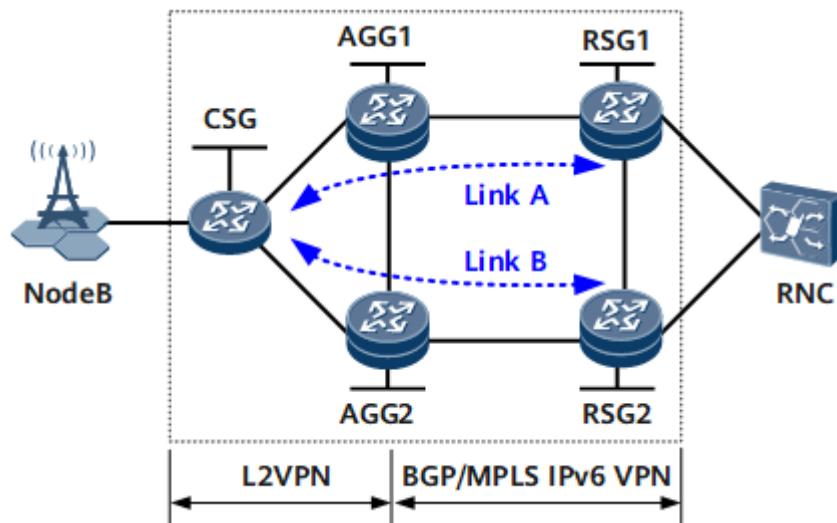
1.1.1.2.12 Configuring an IPv6 Direct Route to Respond to L3VE Interface Status Changes After a Delay

This section describes how to configure an IPv6 direct route to respond to Layer 3 Virtual Ethernet (L3VE) interface status changes after a delay. During traffic switchback after an L3VE interface recovers, this configuration can reduce traffic loss and improve network reliability.

Applicable Environment

As shown in [Figure 1-22](#), a Layer 2 virtual private network (L2VPN) connection is set up between each access aggregation gateway (AGG) and the cell site gateway (CSG) through Layer 2 Virtual Ethernet (L2VE) interfaces, while BGP virtual private network version 4 (VPNv4) peer relationships are set up between AGGs and radio network controller site gateways (RSGs) on a Layer 3 virtual private network (L3VPN). L3VE interfaces are configured on AGGs and VPN instances are bound to the L3VE interfaces, so that the CSG can access the L3VPN. BGP is configured on AGGs to import direct routes between the CSG and AGGs. These direct routes are converted to BGP VPNv6 routes and advertised to RSGs.

Figure 1-22 Networking of an IPv6 direct route responding to L3VE interface status changes after a delay



AGG1 is used as the master device in the preceding figure. In normal cases, RSGs select routes advertised by AGG1 and traffic travels along LinkA. If AGG1 or the CSG-AGG1 link fails, traffic is switched over to LinkB. After AGG1 or the CSG-AGG1 link recovers, the L3VE interface on AGG1 goes from Down to Up, and AGG1 immediately generates an IPv6 direct route destined for the CSG and advertises the route to RSGs. Downstream traffic is then switched over to LinkA. However, AGG1 has not learned the MAC addresses of the base stations yet, so downstream traffic will be lost.

After you configure a cost for an IPv6 direct route and allow the direct route to restore its default cost 0 after a delay, when the L3VE interface on AGG1 goes from Down to Up, the cost of the direct route between the CSG and AGG1 is modified to the configured cost. In this case, RSGs do not select routes advertised

by AGG1 and downstream traffic still travels along LinkB. After the configured delay expires, the cost of the direct route to the CSG is restored to the default value 0. Then, RSGs select routes advertised by AGG1 and downstream traffic is switched over to LinkA. At this time, AGG1 has learned the MAC addresses of the base stations, and downstream traffic loss will be reduced.

Pre-configuration Tasks

Before you configure an IPv6 direct route to respond to L3VE interface status changes after a delay, complete the following task:

- Configure link-layer protocol parameters and IPv6 addresses for interfaces to ensure that the link layer protocol of each interface is up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface virtual-ethernet *interface-number***

A VE interface is created, and the VE interface view is displayed.

Step 3 Run **ve-group *ve-group-id* l3-access**

The VE interface is configured as an L3VE interface and bound to a VE-group.

NOTE

The L2VPN can access the L3VPN only if the L2VE and L3VE interfaces are bound to the same VE-group and reside on the same board.

Step 4 Run **quit**

Return to the system view.

Step 5 Run **interface virtual-ethernet *interface-number*.*subinterface-number***

A VE sub-interface is created, and the VE sub-interface view is displayed.

Step 6 Run **ipv6 enable**

The IPv6 capability is enabled on the interface.

Step 7 Run **direct-route ipv6 degrade-delay *delay-time* degrade-cost *cost***

The direct route is configured to respond to L3VE interface status changes after a delay.

After you run the **direct-route ipv6 degrade-delay** command on an L3VE interface, and the L3VE interface goes from Down to Up, the cost of the L3VE interface's direct route to the CSG is modified to the configured cost. After the configured delay-time expires, the direct route cost is restored to the default value 0.

NOTE

The **direct-route ipv6 degrade-delay** command and the **direct-route ipv6 track pw-state** command cannot be all configured on one L3VE interface.

Step 8 Run commit

The configuration is committed.

----End

Checking the Configurations

If an AGG's L3VE interface goes from Down to Up, run the **display ipv6 routing-table vpn-instance vpn-instance-name [ipv6-address] [verbose]** command on the AGG. The command output shows that the cost of the L3VE interface's IPv6 direct route to the CSG is the configured cost. After the configured *delay-time* expires, run the **display ipv6 routing-table vpn-instance vpn-instance-name [ipv6-address] [verbose]** command on the AGG. The command output shows that the cost of the IPv6 direct route to the L3VE interface is the default value 0.

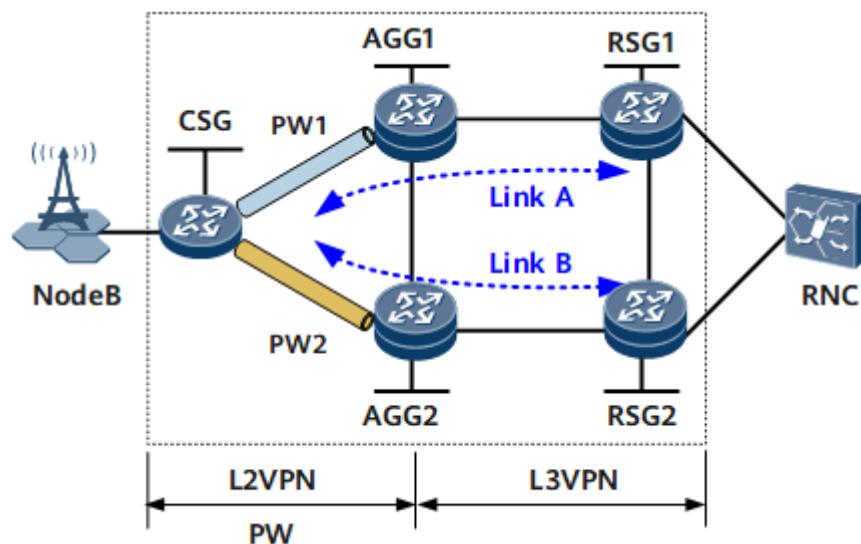
1.1.1.2.13 Configuring the Association Between the IPv4 Direct Route and PW Status

This section describes how to configure the association between the IPv4 direct route and pseudo wire (PW) status. During traffic switchback after the primary PW recovers, this configuration minimizes traffic loss and improves network reliability.

Usage Scenario

On the network shown in [Figure 1-23](#), PWs are set up between the AGGs and the CSG. Border Gateway Protocol (BGP) virtual private network version 4 (VPNv4) peer relationships are set up between the AGGs and RSGs. Layer 3 virtual Ethernet (L3VE) interfaces are configured on the AGGs, and VPN instances are bound to the L3VE interfaces so that the CSG can access the L3VPN. BGP is configured on the AGGs to import IPv4 direct routes between the CSG and AGGs. The AGGs convert these IPv4 direct routes to BGP VPNv4 routes before advertising them to the RSGs.

Figure 1-23 Networking for the association between the IPv4 direct route and PW status



AGG1 functions as the master device. In most cases, the RSGs select routes advertised by AGG1, and traffic travels along Link A. If AGG1 or the CSG-AGG1 link fails, traffic switches over to Link B. After AGG1 or the CSG-AGG1 link recovers, the L3VE interface on AGG1 goes from Down to Up, and AGG1 immediately generates an IPv4 direct route destined for the CSG and advertises the route to the RSGs. Downstream traffic then switches over to Link A. However, PW1 is on standby. As a result, downstream traffic is lost.

After you configure the association between the IPv6 direct route and PW status, when the L3VE interface on AGG1 goes from Down to Up, PW1 does not become the primary PW immediately, and the cost of the IPv6 direct route between the CSG and AGG1 increases. In this case, RSGs do not preferentially select routes advertised by AGG1 and downstream traffic still travels along LinkB. After PW1 between the CSG and AGG1 becomes the primary PW, the cost of the IPv6 direct route between the CSG and AGG1 is restored to the default value 0. Then, RSGs preferentially select routes advertised by AGG1 and downstream traffic is switched over to LinkA. At this time, AGG1 has learned the MAC addresses of the base stations, downstream traffic loss will be reduced after the traffic is switched back.

Pre-configuration Tasks

Before configuring the association between the IPv4 direct route and PW status, configure link-layer protocol parameters and IP addresses for interfaces and ensure that the link layer protocol of each interface is Up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface virtual-ethernet *interface-number***

A VE interface is created, and the VE interface view is displayed.

Step 3 Run **ve-group *ve-group-id* l3-access**

The VE interface is configured as an L3VE interface and bound to a VE-group.



The L2VPN can access the L3VPN only if the L2VE and L3VE interfaces are bound to the same VE-group and reside on the same board.

Step 4 Run **quit**

Return to the system view.

Step 5 Run **interface virtual-ethernet *interface-number*.*subinterface-number***

A VE sub-interface is created, and the VE sub-interface view is displayed.

Step 6 Run **direct-route track *pw-state* degrade-cost *cost***

The association between the IPv4 direct route and PW status is configured.

After you run the **direct-route track *pw-state*** command on an L3VE interface, the system adjusts the cost of the IPv4 direct route to the L3VE interface based on

the PW status. If the PW is on standby, the system modifies the IPv4 direct route cost to the configured value *cost*. If the PW is active, the system restores the IPv4 direct route cost to the default value 0.

 NOTE

The **direct-route track pw-state**, **direct-route degrade-delay**, and **direct-route track vrrp** commands cannot all be configured on one L3VE interface. If you run the three commands on one L3VE interface, the latest configuration overrides the previous ones.

Step 7 Run **commit**

The configuration is committed.

----End

Checking the Configurations

If a PW has recovered but has not become active, run the **display ip routing-table vpn-instance vpn-instance-name [ip-address] [verbose]** command on an AGG. The command output shows that the IPv4 direct route to the L3VE interface has the configured cost. After a PW becomes active, run the **display ip routing-table vpn-instance vpn-instance-name [ip-address] [verbose]** command on an AGG. The command output shows that the IPv4 direct route to the L3VE interface has the default cost 0.

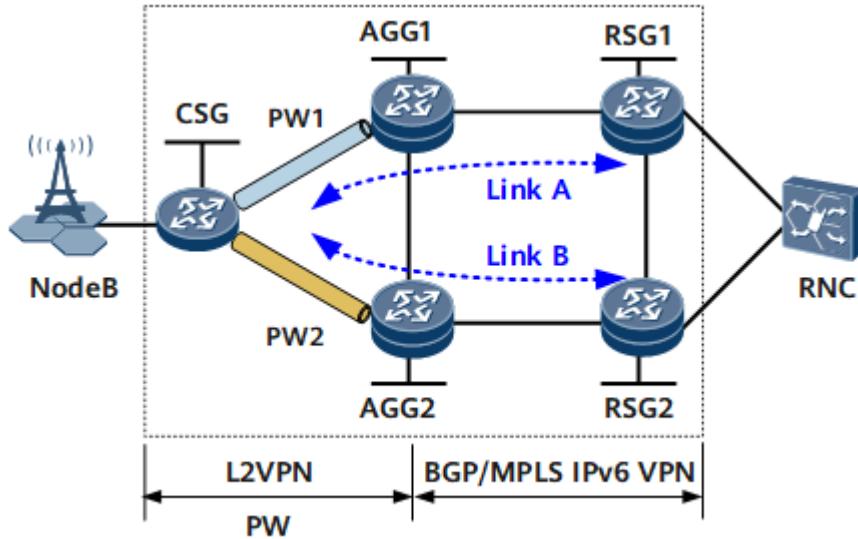
1.1.1.2.14 Configuring the Association Between the IPv6 Direct Route and PW Status

This section describes how to configure the association between the IPv6 direct route and pseudo wire (PW) status. During traffic switchback after the primary PW recovers, this configuration minimizes traffic loss and improves network reliability.

Usage Scenario

As shown in [Figure 1-24](#), PWs are set up between access aggregation gateways (AGGs) and the cell site gateway (CSG), while BGP virtual private network version 6 (VPNv6) peer relationships are set up between AGGs and radio network controller site gateways (RSGs). Layer 3 Virtual Ethernet (L3VE) interfaces are configured on AGGs and VPN instances are bound to the L3VE interfaces, so that the CSG can access the Layer 3 virtual private network (L3VPN). BGP is configured on AGGs to import IPv6 direct routes between the CSG and AGGs. These IPv6 direct routes are converted to BGP VPNv6 routes and advertised to RSGs.

Figure 1-24 Networking of the association between the IPv6 direct route and PW status



AGG1 is used as the master device in the preceding figure. In normal cases, RSGs select routes advertised by AGG1 and traffic travels along LinkA. If AGG1 or the CSG-AGG1 link fails, traffic is switched over to LinkB. After AGG1 or the CSG-AGG1 link recovers, the L3VE interface on AGG1 goes from Down to Up, and AGG1 immediately generates an IPv6 direct route destined for the CSG and advertises the route to RSGs. Downstream traffic is then switched over to LinkA. However, AGG1 has not learned the MAC addresses of the base stations yet and PW1 is standing by, so downstream traffic will be lost.

After you configure the association between the IPv6 direct route and PW status, when the L3VE interface on AGG1 goes from Down to Up, PW1 does not become the primary PW immediately, and the cost of the IPv6 direct route between the CSG and AGG1 increases. In this case, RSGs do not preferentially select routes advertised by AGG1 and downstream traffic still travels along LinkB. After PW1 between the CSG and AGG1 becomes the primary PW, the cost of the IPv6 direct route between the CSG and AGG1 is restored to the default value 0. Then, RSGs preferentially select routes advertised by AGG1 and downstream traffic is switched over to LinkA. At this time, AGG1 has learned the MAC addresses of the base stations, downstream traffic loss will be reduced after the traffic is switched back.

Pre-configuration Tasks

Before configuring the association between the IPv6 direct route and PW status, complete the following tasks:

- Configure link-layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol of each interface is up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface virtual-ethernet interface-number**

A VE interface is created, and the VE interface view is displayed.

Step 3 Run **ve-group ve-group-id l3-access**

The VE interface is configured as an L3VE interface and bound to a VE-group.

 **NOTE**

The L2VPN can access the L3VPN only if the L2VE and L3VE interfaces are bound to the same VE-group and reside on the same board.

Step 4 Run **quit**

Return to the system view.

Step 5 Run **interface virtual-ethernet interface-number.subinterface-number**

A VE sub-interface is created, and the VE sub-interface view is displayed.

Step 6 Run **ipv6 enable**

The IPv6 capability is enabled on the interface.

Step 7 Run **direct-route ipv6 track pw-state degrade-cost cost**

The association between the IPv6 direct route and PW status is configured.

After you run the **direct-route ipv6 track pw-state** command on an L3VE interface, the system adjusts the cost of the IPv6 direct route to the L3VE interface based on the PW status. If the PW is on standby, the system modifies the IPv6 direct route cost to the configured value *cost*. If the PW is active, the system restores the IPv6 direct route cost to the default value 0.

 **NOTE**

The **direct-route ipv6 track pw-state** command and the **direct-route ipv6 degrade-delay** command cannot all be configured on one L3VE interface. If you run the three commands on one L3VE interface, the latest configuration overrides the previous ones.

Step 8 Run **commit**

The configuration is committed.

----End

Checking the Configurations

If a PW has recovered but has not become active, run the **display ipv6 routing-table vpn-instance vpn-instance-name [ipv6-address] [verbose]** command on an AGG. The command output shows that the IPv6 direct route to the L3VE interface has the configured cost. After a PW becomes active, run the **display ipv6 routing-table vpn-instance vpn-instance-name [ipv6-address] [verbose]** command on an AGG. The command output shows that the IPv6 direct route to the L3VE interface has the default cost 0.

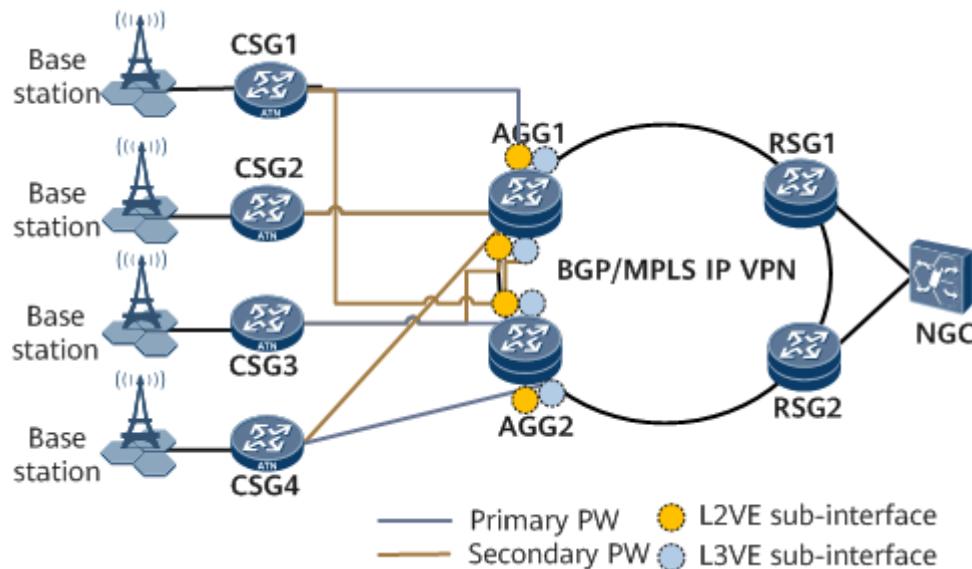
1.1.1.2.15 Configuring AGGs to Load Balance Downstream Traffic

This section describes how to configure access aggregation gateways (AGGs) to load-balance downstream traffic. The load balancing configuration can improve network resource utilization and reliability.

Usage Scenario

Figure 1-25 shows a typical networking for an L2VPN accessing an L3VPN. An L2VPN connection is set up between each AGG and CSG, whereas a BGP VPNv4 peer relationship is set up between each AGG and RSG; L3VE sub-interfaces are configured on each AGG, each L3VE sub-interface is bound to one VPN instance, and the CSGs access the L3VPN. To properly use network resources and improve network reliability, the customer requires that AGG1 forward downstream traffic destined for CSG1 and CSG2 and that AGG2 forward downstream traffic destined for CSG3 and CSG4.

Figure 1-25 Networking for an L2VPN accessing an L3VPN



To meet the preceding requirements, perform the following steps on each AGG:

1. Run the **direct-route cost** or **direct-route ipv6 cost** command to configure a cost for ARP Vlink routes and direct subnet routes on the L3VE sub-interface corresponding to the secondary PW.
2. Configure a static route to each base station and allow the static route to inherit the cost of the route to which the static route recurses.

NOTE

After this configuration is complete, each static route recurses to a direct subnet route on an L3VE interface, and the static route inherits the cost of the direct subnet route. Because the **direct-route cost** or **direct-route ipv6 cost** command is not run on the L3VE sub-interface corresponding to the primary PW, the default cost 0 is used as the cost of the generated direct routes. After a static route recurses to one of the direct routes, the cost of the static route also becomes 0.

3. Import the static routes into the BGP routing table.

 NOTE

After a static route is imported into the BGP routing table, the cost is changed to the BGP route MED. Because the costs of the static routes are different, the BGP route MED values are also different.

After the preceding configurations are complete, each AGG advertises the routes destined for base stations to its VPNV4 peer (RSG). The RSG can then select routes based on MED values. For routes with the same prefix advertised by AGG1 and AGG2, RSGs select the routes advertised by AGG1 as the primary routes to CSG1 and CSG2 and the routes advertised by AGG2 as the primary routes to CSG3 and CSG4. In this manner, downstream traffic is load balanced between AGGs.

Pre-configuration Tasks

Before configuring AGGs to load-balance downstream traffic, configure link-layer protocol parameters and IP addresses for interfaces and ensure that the link layer protocol on each interface is up.

Procedure

Step 1 Set a cost for routes to the directly connected network segment.

1. Run **interface virtual-ethernet *interface-number***

A VE interface is created, and the VE interface view is displayed.

2. Run **ve-group *ve-group-id* l3-access**

The VE interface is configured as an L3VE interface that accesses a BGP/MPLS IP VPN and bound to a VE-group.

 NOTE

L2VPN access can be implemented only if the L2VE and L3VE interfaces are bound to the same VE-group and reside on the same board.

3. Run **quit**

Return to the system view.

4. Run **interface virtual-ethernet *interface-number*.*subinterface-number***

A VE sub-interface is created, and the VE sub-interface view is displayed.

5. Run **direct-route cost *cost* or direct-route ipv6 cost *cost***

A cost is configured for direct subnet routes on the L3VE sub-interface.

6. Run **quit**

Return to the system view.

Step 2 Run **ip route-static vpn-instance *vpn-source-name* destination-address { mask | mask-length } nexthop-address [preference *preference* | tag *tag*] * inherit-cost [description *text*]**

A static route is configured for the VPN instance, and the static route is configured to inherit the cost of the recursive route.

vpn-source-name specifies the name of the VPN instance bound to the L3VE sub-interface. *nexthop-address* specifies the IP address of the interface connecting the base station to the AGG.

Step 3 Perform either of the following operations to configure BGP to import static routes. After the configuration is complete, the costs of the static routes are changed to the MED values of BGP routes.

- To import static routes to the BGP routing table automatically, run the **import-route static** command.
- To import static routes to the BGP routing table manually, run the **network** command.

If you run the **import-route static** command, some unneeded static routes may be imported to the BGP routing table. To import a static route with a specific prefix and mask to the BGP routing table, run the **network** command.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, if the L3VE sub-interface is up, run the **display ip routing-table vpn-instance *vpn-instance-name* [*ip-address*] [verbose]** or **display ipv6 routing-table vpn-instance *vpn-instance-name* [*ip-address*] [verbose]** command on the AGG. The command output shows that the cost of direct subnet routes on the L3VE sub-interface is changed to the configured cost.

1.1.1.2.16 Configuring the Advertisement of Public Network IPv4 ARP Vlink Direct Routes

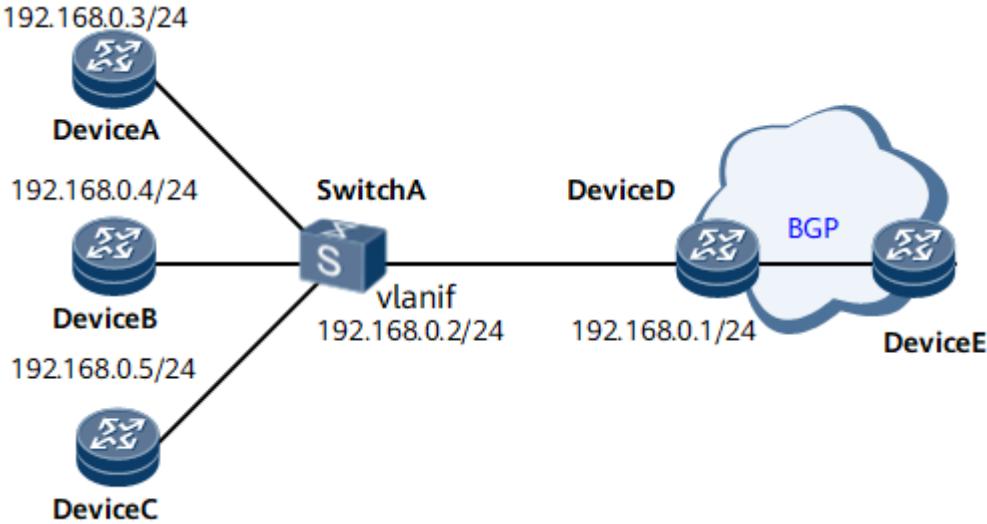
On an IPv4 network, advertising public network IPv4 ARP Vlink direct routes helps with precise control of data traffic.

Usage Scenario

Layer 3 IP forwarding requires physical interfaces to be specified. In a VLAN environment, however, such forwarding cannot be implemented through logical interfaces. Therefore, Layer 3 interfaces of VLAN users need to be obtained through ARP and routing entries containing Layer 3 interface information needs to be generated. Such routes are called IPv4 ARP Vlink direct routes.

As shown in **Figure 1-26**, DeviceD uses logical interfaces to connect to Devices A, B, and C at three sites. DeviceE only needs to communicate with DeviceB, but not with DeviceA or DeviceC. You can configure DeviceD to advertise IPv4 ARP Vlink direct routes and configure a route-policy on DeviceD to filter out routes to the network segment of the VLAN for which the logical interfaces are configured and filter out routes to DeviceA and DeviceC.

Figure 1-26 Networking diagram of advertising IPv4 ARP Vlink direct routes on the public network



Before IPv4 ARP Vlink direct routes are advertised, a route-policy can be configured to filter the advertised routes and only routes that match the route-policy can be advertised. In this manner, data traffic can be precisely controlled.

Perform the following steps on the router on which IPv4 ARP Vlink direct routes need to be advertised.

Pre-configuration Tasks

Before advertising IPv4 ARP Vlink direct routes on the public network, configure parameters of a link layer protocol and assign an IP address to each interface to ensure that the link layer protocol on the interfaces is Up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **arp vlink-direct-route advertise [route-policy route-policy-name | route-filter route-filter-name]**

Advertisement of public network IPv4 ARP Vlink direct routes is configured.

To filter the routes, you can specify **route-policy route-policy-name** or **route-filter route-filter-name**.

NOTE

Currently, apply clauses cannot be used to set route attributes for matched ARP Vlink direct routes.

After the device is enabled to advertise IPv4 ARP Vlink direct routes, such routes can be advertised only after they are imported to the routing table of a dynamic

routing protocol. Perform any of the following operations based on the routing protocol on the router:

- To import IPv4 ARP Vlink direct routes to RIP, run the **import-route direct [cost cost | route-policy route-policy-name]** * command, and then configure advertisement.
- To import IPv4 ARP Vlink direct routes to OSPF, run the **import-route direct [cost cost | route-policy route-policy-name | tag tag | type type]** * command, and then configure advertisement.
- To import IPv4 ARP Vlink direct routes to IS-IS, run the **import-route direct [cost-type { external | internal } | cost cost | tag tag | route-policy route-policy-name | [level-1 | level-2 | level-1-2]]** * command, and then configure advertisement.
- To import IPv4 ARP Vlink direct routes to BGP, run the **import-route direct [med med | route-policy route-policy-name]** * command, and then configure advertisement.

Step 3 (Optional) Run **interface interface-type interface-number**

The interface view is displayed.

Step 4 (Optional) Run **arp vlink-direct-route preference preference-value**

A preference is configured for ARP Vlink direct routes.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring advertisement of public network IPv4 ARP Vlink direct routes, run the **display ip routing-table verbose** command to check public network IPv4 ARP Vlink direct routes.

1.1.1.2.17 Configuring the Advertisement of IPv6 NDP Vlink Direct Routes on the Public Network

On an IPv6 public network, advertising IPv6 neighbor discover protocol (NDP) Vlink direct routes allows precise control of data traffic.

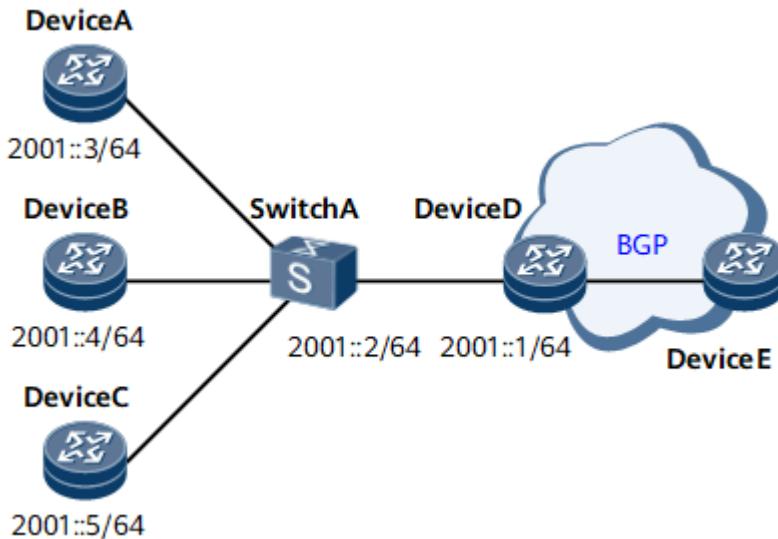
Applicable Environment

IP packets are forwarded through a specified physical interface, but cannot be forwarded through a logical interface. If packets reach a logical interface, the device obtains information about the layer-3 interfaces using IPv6 NDP and generates relevant routing entries. The routes recorded by the routing entries are called IPv6 NDP Vlink direct routes.

As shown in [Figure 1-27](#), Device D uses logical interfaces to connect to Devices A, B, and C at three sites. Device E only needs to communicate with Device B, but not with Device A and Device C. You can configure Device D to advertise IPv6 NDP Vlink direct routes and configure a route-policy on Device D to filter out routes to

the network segment of the VLAN for which the logical interfaces are configured and filter out routes to Device A and Device C.

Figure 1-27 Networking diagram of advertising IPv6 NDP Vlink direct routes on the public network



Before IPv6 NDP Vlink direct routes are advertised, a route-policy can be used to filter the advertised routes and only routes that pass the filtering can be advertised. In this manner, data traffic can be precisely controlled.

Perform the following steps on the router on which IPv6 NDP Vlink direct routes need to be advertised.

Pre-configuration Tasks

Before advertising IPv6 NDP Vlink direct routes on the public network, complete the following task:

- Configuring parameters of a link layer protocol and assigning an IP address to each interface to ensure that the link layer protocol on the interfaces is Up

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **ipv6 nd vlink-direct-route advertise [route-policy route-policy-name | route-filter route-filter-name]**

Advertising IPv6 NDP Vlink direct routes is enabled.

If IPv6 NDP Vlink direct routes have to be advertised, you can specify the parameter **route-policy route-policy-name** or **route-filter route-filter-name** in the **ipv6 nd vlink-direct-route advertise** command to filter IPv4 ARP Vlink direct routes.

 NOTE

At present, **apply** clauses cannot be used to set routing attributes for the NDP Vlink direct routes that match the filtering rules.

Step 3 Run commit

The configuration is committed.

After advertising IPv6 NDP Vlink direct routes is enabled, IPv6 NDP Vlink direct routes can be advertised only if they are imported to dynamic routing protocols. Perform the following steps on the router based on the type of the dynamic routing protocol:

- If RIPng is used, run the **import-route direct [cost cost | route-policy route-policy-name]** * command to import IPv6 NDP Vlink direct routes to RIPng.
- If OSPFv3 is used, run the **import-route direct [cost cost | inherit-cost | route-policy route-policy-name | tag tag | type type]** * command to import IPv6 NDP Vlink direct routes to OSPFv3.
- If IS-IS is used, run the **import-route direct [cost-type { external | internal } | cost cost | tag tag | route-policy route-policy-name | [level-1 | level-2 | level-1-2]]** * command to import IPv6 NDP Vlink direct routes to IS-IS.
- If BGP4+ is used, run the **import-route direct [med med | route-policy route-policy-name]** * command to import IPv6 NDP Vlink direct routes to BGP4+.

----End

Checking the Configurations

Run the **display ipv6 routing-table ipv6-address [prefix-length] [longer-match] [verbose]** command to check information about advertised IPv6 NDP Vlink direct routes on the public network.

1.1.1.2.18 Configuring the Device to Filter ARP or ND Entries When Advertising VPN Vlink Direct Routes

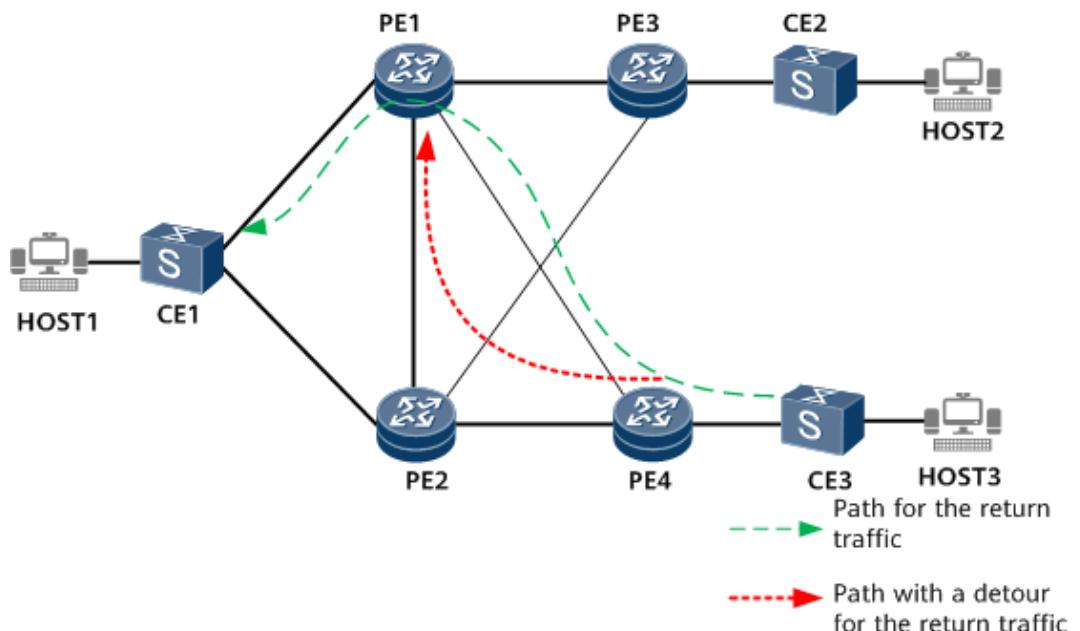
This section describes how to configure a device to filter ARP or ND entries when advertising VPN Vlink direct routes in a scenario where EVPN and L3VPN coexist. This configuration prevents the device from advertising the routes generated based on the ARP or ND entries that are filtered out.

Usage Scenario

During network evolution, some devices may be EVPN-ready, whereas other devices support only traditional functions, such as L3VPN, but not yet EVPN-ready. For example, on the network shown in [Figure 1-28](#), CE1, CE2, and CE3 are access devices at different sites. CE1 is dual-homed to gateway devices PE1 and PE2. PE1, PE2, and PE3 are EVPN-ready gateways, whereas PE4 is still a traditional device. PE1, PE2, and PE3 run EVPN and implement Layer 2 and Layer 3 service interworking for the access devices. In addition, PE1, PE2, and PE4 run traditional L2VPN and L3VPN and implement Layer 2 and Layer 3 service interworking for the access devices. CE1 accesses PE1 and PE2 through Layer 2 sub-interfaces, and the same gateway address is configured for corresponding VBDIF interfaces on

PE1 and PE2. CE1 accesses PE1 and PE2 in dual-homing single-active mode. PE1 is the master device, whereas PE2 is the backup device. PE1 learns the dynamic ARP entry of HOST1 and sends the ARP entry to PE2 through an EVPN IRB route for backup. PE2 then generates a remote dynamic ARP entry of HOST1. PE1 and PE2 generate Vlink routes based on the ARP entries and send VPNv4 host routes to PE4. After receiving these routes, PE4 selects one of them based on route priorities. If the route received from PE2 is selected, traffic travels along the path CE3 -> PE4 -> PE2 -> PE1 -> CE1 (with a detour to PE2) for a long time. Because EVPN does not generate IRB routes based on remote dynamic ARP entries, PE3 can receive the EVPN IRB route only from PE1, and the route is used to guide Layer 3 forwarding. Therefore, traffic from CE2 to CE1 passes through CE2 -> PE3 -> PE1 -> CE1, without a detour to PE2. To prevent a detour, configure PE2 to filter out single-active and redirection ARP entries so that it does not advertise the routes generated based on these entries. In this way, PE4 will not receive the route advertised by PE2, preventing return traffic from detouring to PE2.

Figure 1-28 Coexistence of EVPN dual-homing single-active and L3VPN



Procedure

- Enable the device to filter ARP entries when advertising Vlink routes.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **ip vpn-instance *vpn-instance-name***
A VPN instance is created and its view is displayed.
 - c. Run **ipv4-family**
The IPv4 address family is enabled for the VPN instance, and the VPN instance IPv4 address family view is displayed.
 - d. Run **arp vlink-direct-route advertise-filter single-active-redirect**
The device is enabled to filter ARP entries when advertising Vlink routes.

e. Run **commit**

The configuration is committed.

- Enable the device to filter ND entries when advertising Vlink routes.

a. Run **system-view**

The system view is displayed.

b. Run **ip vpn-instance *vpn-instance-name***

A VPN instance is created and its view is displayed.

c. Run **ipv6-family**

The IPv6 address family is enabled for the VPN instance, and the VPN instance IPv6 address family view is displayed.

d. Run **nd vlink-direct-route advertise-filter single-active-redirect**

The device is enabled to filter ND entries when advertising Vlink routes.

e. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After enabling the device to filter ARP entries when advertising Vlink routes, run the **display ip routing-table all-vpn-instance verbose** command to check information about VPN Vlink direct routes.

After enabling the device to filter ND entries when advertising Vlink routes, run the **display ipv6 routing-table all-vpn-instance verbose** command to check information about VPN Vlink direct routes.

1.1.1.2.19 Configuring the Association Between IPv4 Direct Routes and IPsec Instance Status

The association between IPv4 direct routes and IP security (IPsec) instance status ensures that data encrypted using IPsec can be transmitted to the correct radio network controller site gateway (RSG).

Usage Scenario

In an IP radio access network (IPRAN) scenario, some services require high security. To meet such requirements, cell site gateways (CSGs) encrypt data of these services using IPsec. After the data flows to RSGs (IPsec gateways) through an IPsec tunnel, the RSGs decrypt the data. In most cases, carriers deploy master and backup RSGs and configure the same IP address for the IPsec tunnel interfaces of the master and backup RSGs to improve network reliability.

Without the association between IPv4 direct routes and IPsec instance status, IPv4 direct routes with the same prefix generated on the IPsec tunnel interfaces of the master and backup RSGs share the same default cost (0). As a result, after receiving these routes from the master and backup RSGs, CSGs cannot select an optimal one based on the cost.

With the association between IPv4 direct routes and IPsec instance status:

- If the IPsec instance status is master on an IPsec tunnel interface, the cost of the IPv4 direct routes generated on the interface is 0.
- If the IPsec instance status is backup on an IPsec tunnel interface or the system cannot detect the IPsec instance status, the cost of the IPv4 direct routes generated on the interface is the cost configured on the interface.

After receiving the IPv4 direct routes with the same prefix from the master and backup RSGs, CSGs can select an optimal one based on the cost. Therefore, the CSGs can transmit data encrypted using IPsec to the correct RSG.

Pre-configuration Tasks

Before configuring the association between IPv4 direct routes and IPsec instance status, complete the following tasks:

- Configure link layer protocol parameters and IP addresses for interfaces and ensure that the link layer protocol of each interface is Up.
- Configure IPsec.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface tunnel *interface-number***

A tunnel interface is created, and the tunnel interface view is displayed.

Step 3 Run **tunnel-protocol ipsec**

The encapsulation protocol is set to IPsec on the tunnel interface.

Step 4 Run **ipsec policy *policy-name* service-instance-group *service-group-name* instance *instance-id***

The IPsec policy is applied on the interface.

Step 5 Run **direct-route track ipsec-instance degrade-cost *cost***

The association between IPv4 direct routes and IPsec instance status is configured.



If the IPsec tunnel interface uses the IP address of another interface, the association between the cost of IPv4 direct routes and the IPsec instance status cannot be configured on this IPsec tunnel interface.

The cost of local IPv4 direct routes that are not to be advertised cannot be associated with the IPsec instance status.

Step 6 Run **commit**

The configuration is committed.

----End

Checking the Configurations

After configuring association between IPv4 direct routes and IPsec instance status, run the **display ip routing-table vpn-instance vpn-instance-name [ip-address] [verbose]** command on the RSG to check the information about the IP routing table of the VPN instance or run the **display ip routing-table [ip-address [mask | mask-length] [verbose]]** command to check the information about the IP routing table of the public network instance.

- If the IPsec instance status is master on an IPsec tunnel interface, the cost of the IPv4 direct routes generated on the interface is 0.
- If the IPsec instance status is backup on an IPsec tunnel interface or the system cannot detect the IPsec instance status, the cost of the IPv4 direct routes generated on the interface is the cost configured on the interface.

1.1.1.2.20 Configuring Route or Tunnel Recursion Suppression in Case of Flapping

Usage Scenario

In route or tunnel recursion scenarios, if a route or tunnel flaps frequently, services that depend on the route or tunnel perform recursion frequently, causing frequent route updates. As a result, the CPU usage of the system keeps increasing. To solve the problem, enable route or tunnel recursion suppression in case of flapping. The suppression reduces the route update frequency and the CPU usage. To configure suppression periods for route or tunnel recursion, run the **route recursive-lookup delay** command.

Pre-configuration Tasks

Before configuring route or tunnel recursion suppression in case of flapping, complete the following tasks:

- Configure link layer protocol parameters and IPv4 addresses for interfaces and ensure that the link layer protocol of the interfaces is Up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **undo route recursive-lookup delay disable**

Route or tunnel recursion suppression in case of flapping is enabled.

To disable route or tunnel recursion suppression in case of flapping, run the **route recursive-lookup delay disable** command.

Step 3 (Optional) Run **route recursive-lookup delay start-time start-time increase-time increase-time max-time max-time**

Suppression periods are configured for route or tunnel recursion, including the initial suppression period, incremental suppression period since the second suppression, and the maximum suppression period.

Step 4 Run **commit**

The configuration is committed.

----End

Checking the Configurations

Run the **display current-configuration | include route recursive-lookup delay** command to check configurations.

1.1.1.2.21 Maintaining IP Routes

Maintaining IP routes involves displaying the routing table and routing management module, and debugging the routing management module.

Displaying the Routing Table

Viewing information about the routing table helps locate faults on the network.

Context

It is necessary to check the information in the routing table using **display** commands in order to locate routing problems. The **display** commands can be used in all views. Common commands for displaying routing information are listed as follows:

Procedure

- To check brief information about activated routes in the IP routing table, run the **display ip routing-table** command.
- To check detailed information about the IP routing table, run the **display ip routing-table verbose** command.
- To check limits on the number of routes and prefixes, run the **display ip routing-table limit [all-vpn-instance | vpn-instance vpn-instance-name]** command.
- To check the number of routes, run the **display ip routing-table route-number** command.
- To check the route to the specified destination IP address, run the **display ip routing-table ip-address [mask | mask-length] [longer-match] [verbose]** command.
- To check routes to the specified range of destination IP addresses, run the **display ip routing-table ip-address1 { mask1 | mask-length1 } ip-address2 { mask2 | mask-length2 } [verbose]** command.
- To check the routes discovered by the specified protocol, run the **display ip routing-table protocol { direct | ospf | isis | static | rip | bgp | unr } [inactive | verbose]** command.
- To check general information about the routing table, run the **display ip routing-table statistics** command.
- To check brief information about the VPN routing table, run the **display ip routing-table vpn-instance vpn-instance-name** command.
- To check detailed information about the VPN routing table, run the **display ip routing-table vpn-instance vpn-instance-name verbose** command.

- To check detailed information about the route to the specified destination IP address in the VPN routing table, run the **display ip routing-table vpn-instance vpn-instance-name ip-address [mask | mask-length] [longer-match] [verbose]** command.
- To check detailed information about the routes to the specified network segment in the VPN routing table, run the **display ip routing-table vpn-instance vpn-instance-name ip-address1 { mask1 | mask-length1 } ip-address2 { mask2 | mask-length2 } [verbose]** command.
- To check brief information about activated routes in the IPv6 routing table, run the **display ipv6 routing-table** command.
- To check detailed information about the IPv6 routing table, run the **display ipv6 routing-table verbose** command.
- To check brief information about the IPv6 routing table, run the **display ipv6 routing-table [vpn-instance vpn-instance-name] simple** command.
- To check limits on the number of IPv6 routes and prefixes, run the **display ipv6 routing-table limit [all-vpn-instance | vpn-instance vpn-instance-name]** command.
- To check the number of IPv6 routes, run the **display ipv6 routing-table route-number** command.
- To check the route to the specified destination IPv6 address, run the **display ipv6 routing-table ipv6-address [prefix-length] [longer-match] [verbose]** command.
- To check routes to the specified range of destination IPv6 addresses, run the **display ipv6 routing-table ipv6-address1 [prefix-length1] ipv6-address2 prefix-length2 [verbose]** command.
- To check the IPv6 routes discovered by the specified protocol, run the **display ipv6 routing-table protocol { bgp | direct | isis | ospfv3 | ripng | static | unr } [inactive | verbose]** command.
- To check general information about the IPv6 routing table, run the **display ipv6 routing-table statistics** command.
- To check brief information about the VPN routing table, run the **display ipv6 routing-table vpn-instance vpn-instance-name** command.
- To check detailed information about the VPN routing table, run the **display ipv6 routing-table vpn-instance vpn-instance-name verbose** command.
- To check detailed information about the route to the specified destination IPv6 address in the VPN routing table, run the **display ipv6 routing-table vpn-instance vpn-instance-name ipv6-address [prefix-length] [longer-match] [verbose]** command.
- To check detailed information about the routes to the specified IPv6 network segment in the VPN routing table, run the **display ipv6 routing-table vpn-instance vpn-instance-name ipv6-address1 [prefix-length1] ipv6-address2 prefix-length2 [verbose]** command.

----End

Clearing Routing Table Statistics

Clear historical statistics about protocol routes in the IPv4 and IPv6 routing table. This enables the router to re-collect statistics about protocol routes, facilitating route monitoring and fault location.

Context

NOTICE

Statistics in the IPv4 and IPv6 routing table cannot be restored after you clear them. Therefore, exercise caution when running the command.

Procedure

- To clear statistics about protocol routes in the IPv4 routing table, run the **reset ip routing-table statistics protocol** command in the user view.
- To clear statistics about protocol routes in the IPv6 routing table, run the **reset ipv6 routing-table statistics protocol** command in the user view.

----End

Configuring a Limit on the Number of IPv4 Public Route Prefixes

Configuring a limit on the number of IPv4 public route prefixes can improve system security and reliability.

Context

If the router imports a large number of routes, system performance may be affected when processing services because the routes consume a lot of system resources. To improve system security and reliability, configure a limit on the number of IPv4 public route prefixes. When the number of IPv4 public route prefixes exceeds the limit, an alarm is generated, prompting you to check whether unneeded IPv4 public route prefixes exist.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip prefix-limit number { alert-percent [route-unchanged] | simply-alert }**

A limit is configured on the number of IPv4 public route prefixes.

If you decrease *alert-percent* after the number of IPv4 public route prefixes exceeds *number*, whether the routing table remains unchanged is determined by **route-unchanged**.

- If you specify **route-unchanged** in the command, the routing table remains unchanged.
- If you do not specify **route-unchanged** in the command, the system deletes the routes from the routing table and re-adds routes.

By default, the system deletes the routes from the routing table and re-adds routes.

 NOTE

After the number of IPv4 public route prefixes exceeds the limit, note the following rules:

- If you run the **ip prefix-limit** command to increase *number* or the **undo ip prefix-limit** command to delete the limit, the router relearns IPv4 public route prefixes.
- You can use **display ip routing-table limit [all-vpn-instance | vpn-instance *vpn-instance-name*]** command to display limits on the number of routes and prefixes.
- Direct and static routes can still be added to the IP routing table.

Step 3 Run commit

The configuration is committed.

----End

Configuring a Limit on the Number of IPv6 Public Route Prefixes

Configuring a limit on the number of IPv6 public route prefixes can improve system security and reliability.

Context

If the router imports a large number of routes, system performance may be affected when processing services because the routes consume a lot of system resources. To improve system security and reliability, configure a limit on the number of IPv6 public route prefixes. When the number of IPv6 public route prefixes exceeds the limit, an alarm is generated, prompting you to check whether unneeded IPv6 public route prefixes exist.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **ipv6 prefix-limit *number* { *alert-percent* [**route-unchanged] | **simply-alert** }****

A limit is configured on the number of IPv6 public route prefixes.

If you decrease *alert-percent* after the number of IPv6 public route prefixes exceeds *number*, whether the routing table remains unchanged is determined by **route-unchanged**.

- If you specify **route-unchanged** in the command, the routing table remains unchanged.
- If you do not specify **route-unchanged** in the command, the system deletes the routes from the routing table and re-adds routes.

By default, the system deletes the routes from the routing table and re-adds routes.

 NOTE

After the number of IPv6 public route prefixes exceeds the limit, note the following rules:

- If you run the **ipv6 prefix-limit** command to increase *number* or the **undo ipv6 prefix-limit** command to delete the limit, the router relearns IPv6 public route prefixes.
- You can use **display ipv6 routing-table limit [all-vpn-instance | vpn-instance *vpn-instance-name*]** command to display limits on the number of routes and prefixes.
- Direct and static routes can still be added to the IP routing table.

Step 3 Run commit

The configuration is committed.

----End

Configuring Thresholds for the Number of IPv4 Route Prefixes

This section describes how to configure thresholds (one alarm threshold and one clear alarm threshold) for the number of IPv4 route prefixes on a device. After the thresholds are configured, an alarm is generated when the number of IPv4 route prefixes on the device exceeds the alarm threshold, and the alarm is cleared when the number of IPv4 route prefixes falls below the clear alarm threshold.

Configuring these thresholds facilitates maintenance.

Context

The number of IPv4 route prefixes that can be added to a routing table is limited. If the number exceeds the limit, new prefixes cannot be added to the routing table, which may interrupt services. To address this problem, configure an alarm threshold for the number of IPv4 route prefixes.

An alarm is generated when the following conditions are met:

- The number of IPv4 route prefixes on the device exceeds the alarm threshold.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ip prefix-limit system threshold-alarm upper-limit *upper-limit-value* lower-limit *lower-limit-value*

Two thresholds (one alarm threshold and one clear alarm threshold) are configured for the number of IPv4 route prefixes on the device.

 NOTE

When you configure *upper-limit-value* and *lower-limit-value*, note the following suggestions:

- Set a value less than or equal to 95 for *upper-limit-value*.
- *lower-limit-value* must be less than *upper-limit-value*. Set *lower-limit-value* to a value at least 10 less than *upper-limit-value* to prevent alarms from being frequently generated and cleared due to route flapping.

Step 3 Run commit

The configuration is committed.

----End

Configuring Thresholds for the Number of IPv6 Route Prefixes

This section describes how to configure thresholds (one alarm threshold and one clear alarm threshold) for the number of IPv6 route prefixes on a device. After the thresholds are configured, an alarm is generated when the number of IPv6 route prefixes on the device exceeds the alarm threshold, and the alarm is cleared when the number of IPv6 route prefixes falls below the clear alarm threshold.

Configuring these thresholds facilitates maintenance.

Context

The number of IPv6 route prefixes that can be added to a routing table is limited. If the number exceeds the limit, new prefixes cannot be added to the routing table, which may interrupt services. To address this problem, configure an alarm threshold for the number of IPv6 route prefixes.

An alarm is generated when the following conditions are met:

- The number of IPv6 route prefixes on the device exceeds the alarm threshold.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **ipv6 prefix-limit system threshold-alarm upper-limit *upper-limit-value* lower-limit *lower-limit-value***

Two thresholds (one alarm threshold and one clear alarm threshold) are configured for the number of IPv6 route prefixes on the device.



When you configure *upper-limit-value* and *lower-limit-value*, note the following suggestions:

- Set a value less than or equal to 95 for *upper-limit-value*.
- *lower-limit-value* must be less than *upper-limit-value*. Set *lower-limit-value* to a value at least 10 less than *upper-limit-value* to prevent alarms from being frequently generated and cleared due to route flapping.

Step 3 Run commit

The configuration is committed.

----End

1.1.1.2.22 Configuration Examples for IP Routing Basis

This section describes several configuration examples.

Example for Configuring Public Network IPv4 FRR

With public network IPv4 FRR, traffic can be rapidly switched to a backup link if the primary link fails.

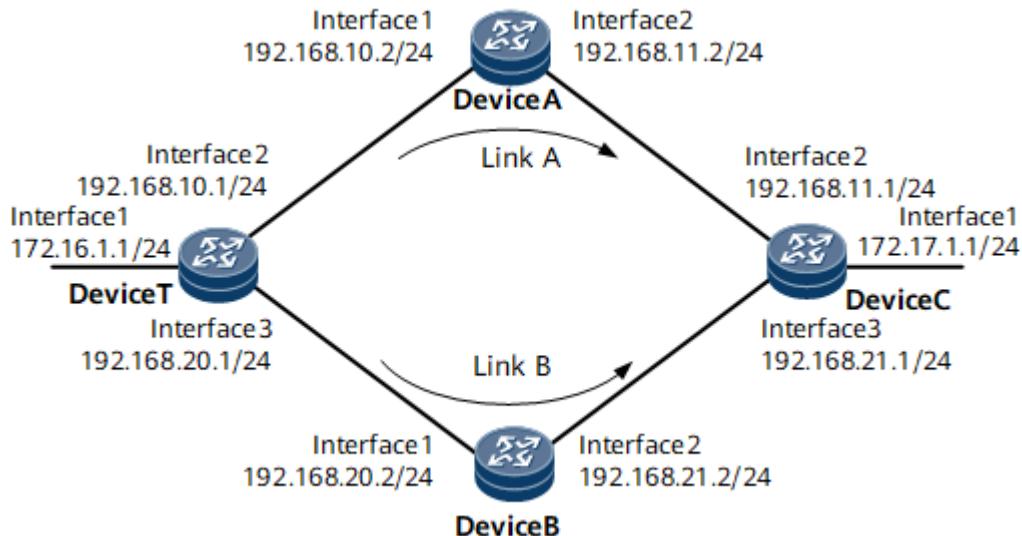
Networking Requirements

On the network shown in [Figure 1-29](#), it is required that the backup outbound interface and backup next hop be configured on Device T to ensure that Link B functions as a backup of Link A. If Link A fails, traffic is rapidly switched to the backup link (Link B).

Figure 1-29 Networking for configuring public network IPv4 FRR



Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Precautions

Before configuring public network IPv4 FRR, there must be at least two routes of different routing protocols but destined for the same IP address.

Configuration Roadmap

The configuration roadmap is as follows:

1. Enable basic OSPF functions on Device T, Device A, and Device C.
2. Configure basic IS-IS functions on Device T, Device B, and Device C.
3. Enable public network IPv4 FRR on Device T, and check the backup outbound interface and backup next hop.
4. Disable IPv4 FRR, and check the backup outbound interface and backup next hop.

Data Preparation

To complete the configuration, you need the following data:

- OSPF process IDs of Device T, Device A, and Device C
- IS-IS area addresses of Device T, Device B, and Device C

Procedure

- Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure OSPF on Device T, Device A, and Device C. For configuration details, see [Configuration Files](#) in this section.
- Step 3** Configure IS-IS on Device T, Device B, and Device C. For configuration details, see [Configuration Files](#) in this section.
- Step 4** Check routing information.

Check the routes to 172.17.1.0 on Device T.

```
<DeviceT> display ip routing-table 172.17.1.0 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Summary Count : 2

Destination: 172.17.1.0/24
  Protocol: OSPF      Process ID: 1
  Preference: 10      Cost: 3
    NextHop: 192.168.10.2   Neighbour: 0.0.0.0
    State: Active Adv     Age: 00h00m07s
      Tag: 0            Priority: low
      Label: NULL        QoSInfo: 0xa98ac7
    IndirectID: 0x40000041
    RelayNextHop: 0.0.0.0   Interface: GigabitEthernet2/0/0
      TunnelID: 0x0          Flags: D
      RouteColor: 0

Destination: 172.17.1.0/24
  Protocol: ISIS       Process ID: 1
  Preference: 15      Cost: 30
    NextHop: 192.168.20.2   Neighbour: 0.0.0.0
    State: Inactive Adv    Age: 00h01m26s
      Tag: 0            Priority: high
      Label: NULL        QoSInfo: 0xa98ac7
    IndirectID: 0x80000081
    RelayNextHop: 0.0.0.0   Interface: GigabitEthernet3/0/0
      TunnelID: 0x0          Flags: 0
      RouteColor: 0
```

In the routing table, you can view that there are two routes to 172.17.1.0/24. The route with 192.168.10.2 as the next hop is optimal because the OSPF route priority is higher than the IS-IS route priority.

- Step 5** Enable public network IPv4 FRR.

Enable IPv4 FRR on Device T.

```
[~DeviceT] ip frr
[*DeviceT] commit
```

Check the backup outbound interface and backup next hop on Device T.

```
<DeviceT> display ip routing-table 172.17.1.0 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Summary Count : 2

Destination: 172.17.1.0/24
    Protocol: OSPF          Process ID: 1
    Preference: 10           Cost: 3
    NextHop: 192.168.10.2   Neighbour: 0.0.0.0
    State: Active Adv       Age: 00h01m36s
    Tag: 0                  Priority: low
    Label: NULL             QoSInfo: 0xa98ac7
    IndirectID: 0x400000041
    RelayNextHop: 0.0.0.0    Interface: GigabitEthernet2/0/0
    TunnelID: 0x0            Flags: D
    RouteColor: 0
    BkNextHop: 192.168.20.2 BkInterface: GigabitEthernet3/0/0
    BkLabel: NULL            SecTunnelID: 0x0
    BkPETunnelID: 0x0        BkPESecTunnelID: 0x0
    BkIndirectID: 0x800000081

Destination: 172.17.1.0/24
    Protocol: ISIS          Process ID: 1
    Preference: 15           Cost: 30
    NextHop: 192.168.20.2   Neighbour: 0.0.0.0
    State: Inactive Adv      Age: 00h02m55s
    Tag: 0                  Priority: high
    Label: NULL             QoSInfo: 0xa98ac7
    IndirectID: 0x800000081
    RelayNextHop: 0.0.0.0    Interface: GigabitEthernet3/0/0
    TunnelID: 0x0            Flags: 0
    RouteColor: 0
```

The routing table shows that the route to 172.17.1.0/24 has the backup outbound interface and backup next hop and that the IS-IS route is the backup route.

Step 6 Verify the configuration.

Simulate a link fault on Device T.

```
[~DeviceT] interface gigabitethernet 2/0/0
[~DeviceT-GigabitEthernet2/0/0] shutdown
[*DeviceT-GigabitEthernet2/0/0] commit
[~DeviceT-GigabitEthernet2/0/0] quit
```

Check the routes to 172.17.1.0/24 on Device T.

```
<DeviceT> display ip routing-table 172.17.1.0 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Summary Count : 1

Destination: 172.17.1.0/24
    Protocol: ISIS          Process ID: 1
    Preference: 15           Cost: 30
    NextHop: 192.168.20.2   Neighbour: 0.0.0.0
    State: Active Adv       Age: 00h57m30s
    Tag: 0                  Priority: high
    Label: NULL             QoSInfo: 0xa98ac7
    IndirectID: 0x800000081
    RelayNextHop: 0.0.0.0    Interface: GigabitEthernet3/0/0
    TunnelID: 0x0            Flags: D
    RouteColor: 0
```

The preceding command output shows that traffic is switched to Link B after Link A fails.

----End

Configuration Files

- Device T configuration file

```
#  
sysname DeviceT  
#  
ip frr  
#  
isis 1  
network-entity 10.0000.0000.0001.00  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 172.16.1.1 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.10.1 255.255.255.0  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 192.168.20.1 255.255.255.0  
isis enable 1  
#  
ospf 1  
area 0.0.0  
network 192.168.10.0 0.0.0.255  
area 0.0.1  
network 172.16.1.0 0.0.0.255  
#  
return
```

- Device A configuration file

```
#  
sysname DeviceA  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.10.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.11.2 255.255.255.0  
#  
ospf 1  
area 0.0.0  
network 192.168.10.0 0.0.0.255  
network 192.168.11.0 0.0.0.255  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
isis 1  
network-entity 10.0000.0000.0002.00  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.20.2 255.255.255.0  
isis enable 1  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.21.2 255.255.255.0  
isis enable 1  
#  
return
```

- Device C configuration file

```
#  
sysname DeviceC  
#  
isis 1  
network-entity 10.0000.0000.0003.00  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 172.17.1.1 255.255.255.0  
isis enable 1  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.11.1 255.255.255.0  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 192.168.21.1 255.255.255.0  
isis enable 1  
#  
ospf 1  
area 0.0.0  
network 192.168.11.0 0.0.0.255  
network 192.168.21.0 0.0.0.255  
area 0.0.2  
network 172.17.1.0 0.0.0.255  
#  
return
```

Example for Configuring Public Network IPv6 FRR

With public network IPv6 FRR, traffic can be rapidly switched to a backup link if the primary link fails.

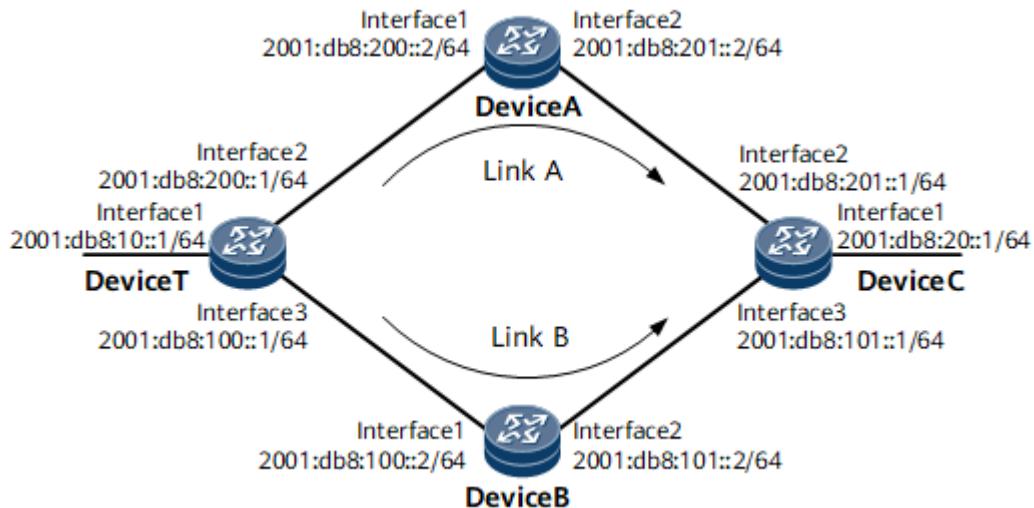
Networking Requirements

On the network shown in [Figure 1-30](#), it is required that the backup outbound interface and backup next hop must be configured on Device T so that link B functions as the backup of link A. If link A fails, traffic is rapidly switched to the backup link (Link B).

Figure 1-30 Networking for configuring public network IPv6 FRR



Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Precautions

Before configuring public network IPv6 FRR, there must be at least two routes of different routing protocols but destined for the same IPv6 address.

Configuration Roadmap

The configuration roadmap is as follows:

1. Enable OSPFv3 on Device T, Device A, and Device C.
2. Enable IPv6 IS-IS on Device T, Device B, and Device C.
3. Enable public network IPv6 FRR on Device T, and then check information about the backup outbound interface and backup next hop.
4. Disable IPv6 FRR, and then check information about the backup outbound interface and backup next hop.

Data Preparation

To complete the configuration, you need the following data:

- OSPFv3 process IDs of Device T, Device A, and Device C (OSPFv3 process ID is 1)
- IPv6 IS-IS area addresses of Device T, Device B, and Device C

Procedure

- Step 1** Configure an IPv6 address for each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure OSPFv3 on Device T, Device A, and Device C. For configuration details, see [Configuration Files](#) in this section.
- Step 3** Configure IPv6 IS-IS on Device T, Device B, and Device C. For configuration details, see [Configuration Files](#) in this section.

Step 4 Check routing information.

On Device T, check the routes to 2001:db8:20::1.

```
<DeviceT> display ipv6 routing-table 2001:db8:20::1 64 verbose
Routing Table : _public_
Summary Count : 2

Destination : 2001:db8:20:: PrefixLength : 64
NextHop   : 2001:db8:200::2 Preference  : 10
Neighbour  : :: ProcessID   : 1
Label      : NULL Protocol    : OSPFv3
State      : Active Adv Cost        : 3
Entry ID   : 0 EntryFlags   : 0x00000000
Reference Cnt: 0 Tag          : 0
IndirectID : 0x69 Age          : 269sec
RelayNextHop : :: TunnelID    : 0x0
Interface   : gigabitethernet 2/0/0 Flags        : D

Destination : 2001:db8:20:: PrefixLength : 64
NextHop   : 2001:db8:100::2 Preference  : 15
Neighbour  : :: ProcessID   : 1
Label      : NULL Protocol    : ISIS
State      : Inactive Adv Cost        : 30
Entry ID   : 0 EntryFlags   : 0x00000000
Reference Cnt: 0 Tag          : 0
IndirectID : 0xb5 Age          : 201sec
RelayNextHop : :: TunnelID    : 0x0
Interface   : gigabitethernet 3/0/0 Flags        : 0
```

The preceding command output shows that there are two routes to 2001:db8:20::1/64 and that the route with 2001:db8:200::2 as the next hop is optimal because the OSPFv3 route priority is higher than the IPv6 IS-IS route priority.

Step 5 Enable public network IPv6 FRR.

Enable IPv6 FRR on Device T.

```
[~DeviceT] ipv6 frr
[*DeviceT] commit
```

Check information about the backup outbound interface and backup next hop on Device T.

```
<DeviceT> display ipv6 routing-table 2001:db8:20::1 64 verbose
Routing Table : _public_
Summary Count : 2

Destination : 2001:db8:20:: PrefixLength : 64
NextHop   : 2001:db8:200::2 Preference  : 10
Neighbour  : :: ProcessID   : 1
Label      : NULL Protocol    : OSPFv3
State      : Active Adv Cost        : 3
Entry ID   : 0 EntryFlags   : 0x00000000
Reference Cnt: 0 Tag          : 0
IndirectID : 0x69 Age          : 553sec
RelayNextHop : :: TunnelID    : 0x0
Interface   : gigabitethernet 2/0/0 Flags        : D
BkNextHop  : 2001:db8:100::2 BkInterface  : gigabitethernet 3/0/0
BkLabel    : NULL BkTunnelID  : 0x0
BkPETunnelID: 0xb5 BkIndirectID : 0xb5

Destination : 2001:db8:20:: PrefixLength : 64
NextHop   : 2001:db8:100::2 Preference  : 15
Neighbour  : :: ProcessID   : 1
Label      : NULL Protocol    : ISIS
State      : Inactive Adv Cost        : 30
```

Entry ID : 0	EntryFlags : 0x00000000
Reference Cnt: 0	Tag : 0
IndirectID : 0xb5	Age : 485sec
RelayNextHop ::	TunnelID : 0x0
Interface : gigabitethernet 3/0/0	Flags : 0

The preceding command output shows that the route to 2001:db8:20::1/64 has a backup outbound interface and a backup next hop and that the IPv6 IS-IS route is the backup route.

Step 6 Verify the configuration.

Simulate a link fault on Device T.

```
[~DeviceT] interface gigabitethernet 2/0/0
[~DeviceT-GigabitEthernet2/0/0] shutdown
[*DeviceT-GigabitEthernet2/0/0] commit
[~DeviceT-GigabitEthernet2/0/0] quit
```

On Device T, check the routes to 2001:db8:20::1/64.

```
<DeviceT> display ipv6 routing-table 2001:db8:20::1 64 verbose
Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:20::          PrefixLength : 64
NextHop    : 2001:db8:100:2        Preference   : 15
Neighbour   ::                      ProcessID   : 1
Label      : NULL                  Protocol    : ISIS
State       : Active Adv          Cost        : 30
Entry ID   : 0                     EntryFlags  : 0x00000000
Reference Cnt: 0                  Tag         : 0
IndirectID : 0xb5                Age         : 1279sec
RelayNextHop ::                   TunnelID   : 0x0
Interface   : gigabitethernet 3/0/0 Flags       : D
```

The preceding command output shows that traffic has been switched to link B.

----End

Configuration Files

- Device T configuration file

```
#
sysname DeviceT
#
ipv6 frr
#
isis 1
is-level level-1
ipv6 enable topology ipv6
network-entity 10.0000.0000.0001.00
#
ospfv3 1
router-id 1.1.1.1
area 0.0.0
area 0.0.0.1
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:10::1/64
ospfv3 1 area 0.0.0.1
isis ipv6 enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
```

```
ipv6 enable
ipv6 address 2001:db8:200::1/64
ospfv3 1 area 0.0.0.
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:100::1/64
isis enable 1
isis ipv6 enable 1
return
```

- Device A configuration file

```
# 
sysname DeviceA
#
ospfv3 1
router-id 2.2.2.2
area 0.0.0.
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:200::2/64
ospfv3 1 area 0.0.0.
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:201::2/64
ospfv3 1 area 0.0.0.
return
```

- Device B configuration file

```
# 
sysname DeviceB
#
isis 1
is-level level-1
ipv6 enable topology ipv6
network-entity 10.0000.0000.0002.00
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:100::2/64
isis ipv6 enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:101::2/64
isis ipv6 enable 1
#
return
```

- Device C configuration file

```
# 
sysname DeviceC
#
isis 1
is-level level-1
ipv6 enable topology ipv6
network-entity 10.0000.0000.0003.00
#
ospfv3 1
router-id 1.1.1.1
area 0.0.0.
area 0.0.0.2
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:20::1/64
isis enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:201::1/64
ospfv3 1 area 0.0.0.2
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:101::1/64
isis enable 1
isis ipv6 enable 1
#
return
```

Example for Associating Direct Routes with a VRRP Group

Associating direct routes with a Virtual Router Redundancy Protocol (VRRP) group prevents user-to-network traffic and network-to-user traffic that traverse the VRRP group from being transmitted on different paths.

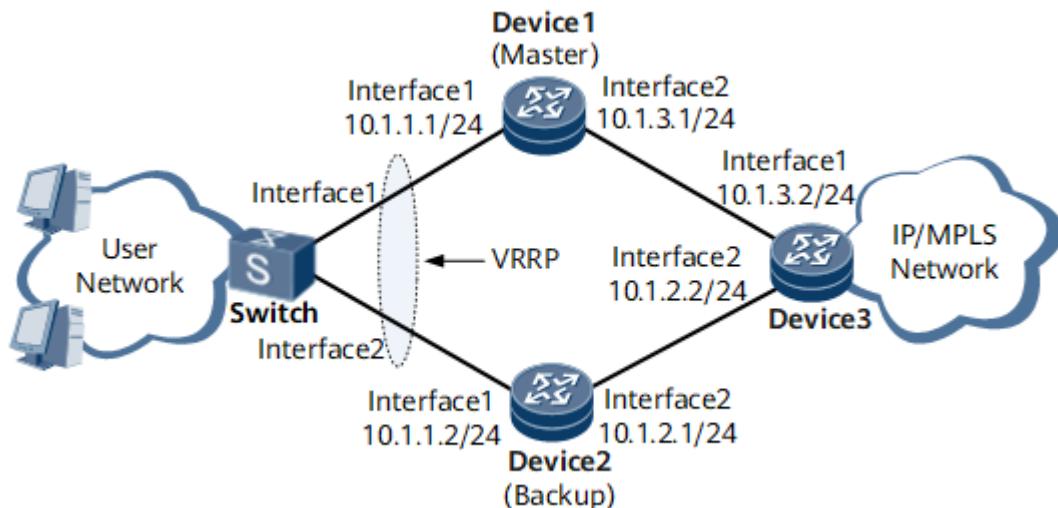
Networking Requirements

On the network shown in [Figure 1-31](#), the VRRP group consisting of Device1 and Device2 functions as the user-side gateway. User-to-network traffic is forwarded through the master device (Device1) of the VRRP group. In addition, the Open Shortest Path First (OSPF) protocol runs on Device 1, Device 2, and Device 3 so that these devices can communicate with one another at the IP layer. Network-to-user traffic travels on the path that OSPF selects. Device 3 has two equal-cost routes destined for the user network segment 10.1.1.0/24 that OSPF selects. Therefore, the two routes load-balance the network-to-user traffic. User-to-network traffic and network-to-user traffic travel on different paths.

Figure 1-31 Association between direct routes and a VRRP group

 **NOTE**

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 1/0/1, respectively.



To resolve this problem, associate direct routes with the VRRP group on VRRP-enabled interfaces of Device1 and Device2 to increase the cost of the direct route to the virtual IP network segment of the VRRP group, import the direct route to OSPF, and configure OSPF to inherit the costs of the imported external routes. Because the cost of the OSPF route passing through Device1 is smaller, Device3 preferentially selects this route to forward downstream traffic. In this manner, upstream and downstream traffic is transmitted along the same path.

Precautions

During the configuration, pay attention to the following points:

- OSPF cannot run on Device 1's and Device 2's VRRP-enabled interfaces. If both VRRP and OSPF run on Device 1's and Device 2's interfaces, OSPF cannot inherit the costs from the imported direct routes after the association.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure a VRRP group on Device 1 and Device 2.
2. Configure OSPF on Device 1, Device 2, and Device 3.
3. Associate direct routes with the VRRP group on Device 1 and Device 2.
4. Enable OSPF to import direct routes on Device 1 and Device 2 and to inherit the costs of the imported direct routes.

Data Preparation

To complete the configuration, you need the following data:

- VLAN ID 10 configured on the Switch
- VRRP ID 1 and a virtual IP address 10.1.1.111 of the VRRP group
- Device 1's VRRP priority 120
- Cost of direct routes to the virtual IP network segment on Device 2 (300)

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration File](#) in this section.

Step 2 Configure a VRRP group.

```
# Configure a VLAN with VLAN ID 10 on the Switch. Add GE 1/0/0 and GE 1/0/1 to VLAN 10 so that the Switch transparently transmits VRRP packets sent by Device 1 or Device 2.
```

```
<Switch> system-view
[~Switch] interface gigabitethernet 1/0/0
[~Switch-GigabitEthernet1/0/0] portswitch
[*Switch-GigabitEthernet1/0/0] commit
[*Switch-GigabitEthernet1/0/0] quit
[*Switch] interface gigabitethernet 1/0/1
[~Switch-GigabitEthernet1/0/1] portswitch
[*Switch-GigabitEthernet1/0/1] commit
[~Switch-GigabitEthernet1/0/1] quit
[*Switch] vlan 10
[*Switch-vlan10] port gigabitethernet 1/0/0
[*Switch-vlan10] port gigabitethernet 1/0/1
[*Switch-vlan10] commit
[~Switch-vlan10] quit
```

```
# Create VRRP group 1 on Device 1, and set the priority of Device 1 in the VRRP group to 120 so that Device 1 functions as the master device.
```

```
<Device1> system-view
[~Device1] interface gigabitethernet 1/0/0
[*Device1-GigabitEthernet1/0/0] vrrp vrid 1 virtual-ip 10.1.1.111
[*Device1-GigabitEthernet1/0/0] vrrp vrid 1 priority 120
[*Device1-GigabitEthernet1/0/0] commit
[~Device1-GigabitEthernet1/0/0] quit
```

```
# Create VRRP group 1 on Device 2
```

```
<Device2> system-view
[~Device2] interface gigabitethernet 1/0/0
[*Device2-GigabitEthernet1/0/0] vrrp vrid 1 virtual-ip 10.1.1.111
[*Device2-GigabitEthernet1/0/0] commit
[~Device2-GigabitEthernet1/0/0] quit
```

Step 3 Configure OSPF.

```
# Configure OSPF on Device 1.
```

```
[~Device1] ospf 1
[*Device1-ospf-1] area 0
[*Device1-ospf-1-area-0.0.0.0] network 10.1.3.0 0.0.0.255
[*Device1-ospf-1-area-0.0.0.0] commit
[~Device1-ospf-1-area-0.0.0.0] quit
[~Device1-ospf-1] quit
```

```
# Configure OSPF on Device 2.
```

```
[~Device2] ospf 1
[*Device2-ospf-1] area 0
[*Device2-ospf-1-area-0.0.0.0] network 10.1.2.0 0.0.0.255
[*Device2-ospf-1-area-0.0.0.0] commit
[~Device2-ospf-1-area-0.0.0.0] quit
[~Device2-ospf-1] quit
```

```
# Configure OSPF on Device 3.
```

```
<Device3> system-view
[~Device3] ospf 1
```

```
[*Device3-ospf-1] area 0
[*Device3-ospf-1-area-0.0.0] network 10.1.2.0 0.0.0.255
[*Device3-ospf-1-area-0.0.0] network 10.1.3.0 0.0.0.255
[*Device3-ospf-1-area-0.0.0] commit
[~Device3-ospf-1-area-0.0.0] quit
[~Device3-ospf-1] quit
```

Step 4 Associate direct routes with the VRRP group.

Configure Device1.

```
[~Device1] interface gigabitethernet 1/0/0
[*Device1-GigabitEthernet1/0/0] direct-route track vrrp vrid 1 degrade-cost 300
[*Device1-GigabitEthernet1/0/0] commit
[~Device1-GigabitEthernet1/0/0] quit
```

Configure Device2.

```
[~Device2] interface gigabitethernet 1/0/0
[*Device2-GigabitEthernet1/0/0] direct-route track vrrp vrid 1 degrade-cost 300
[*Device2-GigabitEthernet1/0/0] commit
[~Device2-GigabitEthernet1/0/0] quit
```

Step 5 Configure OSPF to import direct routes.

Configure OSPF on Device 1.

```
[~Device1] ospf 1
[*Device1-ospf-1] import-route direct
[*Device1-ospf-1] default cost inherit-metric
[*Device1-ospf-1] commit
[~Device1-ospf-1] quit
```

Configure OSPF on Device 2.

```
[~Device2] ospf 1
[*Device2-ospf-1] import-route direct
[*Device2-ospf-1] default cost inherit-metric
[*Device2-ospf-1] commit
[~Device2-ospf-1] quit
```

Step 6 Check the configuration.

Run the **display ip routing-table** command on Device2 to view information about the IP routing table. The command output shows that the cost of the direct route to the VRRP virtual IP network segment is 300.

```
<Device2> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 12      Routes : 12
Destination/Mask Proto Pre Cost     Flags NextHop       Interface
10.1.1.0/24 Direct 0   300        D 10.1.1.2      GigabitEthernet1/0/0
10.1.1.2/32 Direct 0   0          D 127.0.0.1    GigabitEthernet1/0/0
10.1.1.111/32 O_ASE 150 0        D 10.1.2.2      GigabitEthernet1/0/1
10.1.1.255/32 Direct 0   0          D 127.0.0.1    GigabitEthernet1/0/0
10.1.2.0/24 Direct 0   0          D 10.1.2.1      GigabitEthernet1/0/1
10.1.2.1/32 Direct 0   0          D 127.0.0.1    GigabitEthernet1/0/1
10.1.2.255/32 Direct 0   0          D 127.0.0.1    GigabitEthernet1/0/1
10.1.3.0/24 OSPF   10  2          D 10.1.2.2      GigabitEthernet1/0/1
127.0.0.0/8 Direct 0   0          D 127.0.0.1    InLoopBack0
127.0.0.1/32 Direct 0   0          D 127.0.0.1    InLoopBack0
127.255.255.255/32 Direct 0   0          D 127.0.0.1    InLoopBack0
255.255.255.255/32 Direct 0   0          D 127.0.0.1    InLoopBack0
```

Run the **display ip routing-table 10.1.1.0** command on Device3 to view information about the route destined for the network segment. The command

output shows that Device3 selects the route with next-hop address 10.1.3.1 (passing through VRRP master device Device1) for downstream traffic to the user.

```
<Device3> display ip routing-table 10.1.1.0
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table : _public_
Summary Count : 1
```

Destination/Mask	Proto	Pre	Cost	Flags	NextHop	Interface
10.1.1.0/24	O_ASE	150	0	D	10.1.3.1	GigabitEthernet1/0/0

----End

Configuration Files

- Switch configuration file

```
#
sysname Switch
#
vlan batch 10
#
interface GigabitEthernet1/0/0
portswitch
undo shutdown
port default vlan 10
#
interface GigabitEthernet1/0/1
portswitch
undo shutdown
port default vlan 10
#
```

- Device 1 configuration file

```
#
sysname Device1
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
vrrp vrid 1 virtual-ip 10.1.1.111
vrrp vrid 1 priority 120
direct-route track vrrp vrid 1 degrade-cost 300
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.3.1 255.255.255.0
#
ospf 1
default cost inherit-metric
import-route direct
area 0.0.0
network 10.1.3.0 0.0.0.255
#
```

- Device 2 configuration file

```
#
sysname Device2
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
vrrp vrid 1 virtual-ip 10.1.1.111
direct-route track vrrp vrid 1 degrade-cost 300
#
interface GigabitEthernet1/0/1
undo shutdown
```

```
ip address 10.1.2.1 255.255.255.0
#
ospf 1
  default cost inherit-metric
  import-route direct
  area 0.0.0
    network 10.1.2.0 0.0.0.255
#
#
```

- Device 3 configuration file

```
#
sysname Device3
#
interface GigabitEthernet1/0/0
  undo shutdown
  ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet1/0/1
  undo shutdown
  ip address 10.1.2.2 255.255.255.0
#
ospf 1
  area 0.0.0
    network 10.1.2.0 0.0.0.255
    network 10.1.3.0 0.0.0.255
#
#
```

Example for Importing IPv4 ARP Vlink Direct Routes to BGP

By importing IPv4 ARP Vlink direct routes to BGP, you can enable the remote device to obtain information about detailed routes in the VLAN, allowing precise control of data traffic.

Networking Requirements

As networks develop, the VLAN technology is widely used. If a user outside a VLAN needs to communicate with users within the VLAN, advertising routes destined for the network segment of the VLAN can achieve this purpose. When users outside the VLAN need to know the IPv4 ARP Vlink direct routes of the VLAN, and apply different traffic policies to routes of the VLAN users, advertising the routes destined for the network segment of the VLAN cannot meet this requirement.

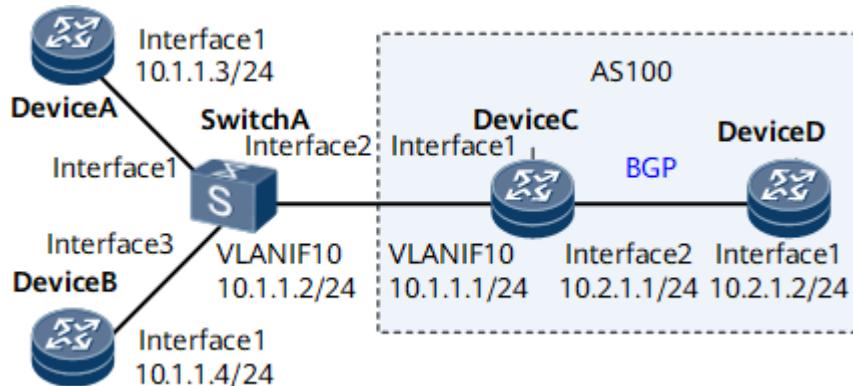
In this case, you can enable the function of IPv4 ARP Vlink direct route advertisement. As shown in [Figure 1-32](#), Device C is connected to two VLAN sites through VLANIF interfaces. Device D communicates with Device B, but not with Device A. To meet the communication requirement, you can enable the function of IPv4 ARP Vlink direct route advertisement on Device C, and use a route-policy to filter out the routes to the network segment of the VLAN and the route to Device A.

Figure 1-32 Networking diagram of importing IPv4 ARP Vlink direct routes to BGP



NOTE

Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Create VLANIF interfaces on Switch A and Device C and assign IP addresses to the VLANIF interfaces, and ensure that Device A, Device B, Switch A, and Device C can communicate with each other.
2. Enable BGP on Device C and Device D, ensuring that Device C and Device D are able to advertise IP routes to each other.
3. Enable the function of IPv4 ARP Vlink direct route advertisement on Device C.
4. Configure a route-policy on Device C, allowing routes only from Device B to pass through.
5. Enable BGP on Device C to import direct routes, and use the route-policy to import routes only from Device B.
6. Associate BGP with the route-policy on Device C to filter out the network segment route of the VLAN so that Device D cannot learn the network segment route and can communicate with VLAN users based only on IPv4 ARP Vlink direct routes.

Data Preparation

To complete the configuration, you need the following data:

- ID of the VLAN in which Switch A and Device C reside (the VLAN ID is 10 in this example)
- Router IDs and AS numbers of Devices C and D (router ID of Device C is 3.3.3.3 and router ID of Device D is 4.4.4.4, and Devices C and D are in AS 100 in this example)
- Route-policy used to filter direct routes (the route-policy is policy1 in this example)
- Route-policy used to advertise BGP routes on Device C (the route-policy is policy2 in this example)

Procedure

Step 1 Configure an IP address for each interface.

Configure Device A.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
[*HUAWEI] commit
[~DeviceA] interface GigabitEthernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] undo shutdown
[*DeviceA-GigabitEthernet1/0/0] ip address 10.1.1.3 24
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

Configure Device B.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[~DeviceB] interface GigabitEthernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] undo shutdown
[*DeviceB-GigabitEthernet1/0/0] ip address 10.1.1.4 24
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

Configure Device C.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceC
[*HUAWEI] commit
[~DeviceC] interface GigabitEthernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] undo shutdown
[*DeviceC-GigabitEthernet2/0/0] ip address 10.2.1.1 24
[*DeviceC-GigabitEthernet2/0/0] commit
[~DeviceC-GigabitEthernet2/0/0] quit
```

Configure Device D.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceD
[*HUAWEI] commit
[~DeviceD] interface GigabitEthernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] undo shutdown
[*DeviceD-GigabitEthernet1/0/0] ip address 10.2.1.2 24
[*DeviceD-GigabitEthernet1/0/0] commit
[~DeviceD-GigabitEthernet1/0/0] quit
```

Step 2 Configure basic VLAN functions. Create VLANIF 10 on Switch A and Device C and assign IP addresses to the VLANIF interfaces.

Configure Switch A.

```
<HUAWEI> system-view
[~HUAWEI] sysname SwitchA
[*HUAWEI] commit
[~SwitchA] vlan 10
[*SwitchA-vlan10] quit
[*SwitchA] interface GigabitEthernet 1/0/0
[*SwitchA-GigabitEthernet1/0/0] portswitch
[*SwitchA-GigabitEthernet1/0/0] undo shutdown
[*SwitchA-GigabitEthernet1/0/0] port link-type access
[*SwitchA-GigabitEthernet1/0/0] port default vlan 10
[*SwitchA-GigabitEthernet1/0/0] quit
[*SwitchA] interface GigabitEthernet 2/0/0
[*SwitchA-GigabitEthernet2/0/0] portswitch
[*SwitchA-GigabitEthernet2/0/0] undo shutdown
[*SwitchA-GigabitEthernet2/0/0] port link-type access
[*SwitchA-GigabitEthernet2/0/0] port default vlan 10
```

```
[*SwitchA-GigabitEthernet2/0/0] quit
[*SwitchA] interface GigabitEthernet 3/0/0
[*SwitchA-GigabitEthernet3/0/0] portswitch
[*SwitchA-GigabitEthernet3/0/0] undo shutdown
[*SwitchA-GigabitEthernet3/0/0] port link-type access
[*SwitchA-GigabitEthernet3/0/0] port default vlan 10
[*SwitchA-GigabitEthernet3/0/0] quit
[*SwitchA] interface Vlanif 10
[*SwitchA-Vlanif10] ip address 10.1.1.2 24
[*SwitchA-Vlanif10] commit
[~SwitchA-Vlanif10] quit
```

Configure Device C.

```
[~DeviceC] vlan 10
[~DeviceC-vlan10] quit
[*DeviceC] interface GigabitEthernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] portswitch
[*DeviceC-GigabitEthernet1/0/0] undo shutdown
[*DeviceC-GigabitEthernet1/0/0] port link-type access
[*DeviceC-GigabitEthernet1/0/0] port default vlan 10
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface Vlanif 10
[*DeviceC-Vlanif10] ip address 10.1.1.1 24
[*DeviceC-Vlanif10] commit
[~DeviceC-Vlanif10] quit
```

Step 3 Configure BGP between Device C and Device D.

Configure Device C.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.2.1.2 as-number 100
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

Configure Device D.

```
[~DeviceD] bgp 100
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 10.2.1.1 as-number 100
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

Step 4 Configure BGP on Device C and import direct routes to BGP. Then view the routing tables of Devices C and D.

Configure Device C.

```
[~DeviceC] bgp 100
[~DeviceC-bgp] import-route direct
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

Display the BGP routing table of Device C.

```
[~DeviceC] display bgp routing-table
BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 9
      Network          NextHop        MED     LocPrf  PrefVal Path/Ogn
      *> 10.1.1.0/24    0.0.0.0       0         0      ? 
      *> 10.1.1.1/32    0.0.0.0       0         0      ? 
```

```
*> 10.1.1.2/32    0.0.0.0      0          0      ?
*> 10.2.1.0/24    0.0.0.0      0          0      ?
*> 10.2.1.1/32    0.0.0.0      0          0      ?
*> 127.0.0.0      0.0.0.0      0          0      ?
*> 127.0.0.1/32   0.0.0.0      0          0      ?
```

Display the BGP routing table of Device D.

```
[~DeviceD] display bgp routing-table
BGP Local router ID is 4.4.4.4
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
Network          NextHop        MED     LocPrf  PrefVal Path/Ogn
*>i 10.1.1.0/24  10.2.1.1      0       100     0      ?
i 10.2.1.0/24    10.2.1.1      0       100     0      ?
```

You can see that Device D has not learned the two IPv4 ARP Vlink direct routes 10.1.1.3/32 and 10.1.1.4/32.

Step 5 Enable the function of IPv4 ARP Vlink direct route advertisement on Device C and configure the route-policy **policy1** to filter out the routes to the network segment of the VLAN and the IPv4 ARP Vlink direct route from Device A, 10.1.1.3/32.

Configure Device C.

```
[~DeviceC] ip ip-prefix prefix1 permit 10.1.1.4 32
[*DeviceC] route-policy policy1 permit node 10
[*DeviceC-route-policy] if-match ip-prefix prefix1
[*DeviceC-route-policy] quit
[*DeviceC] arp vlink-direct-route advertise route-policy policy1
[*DeviceC] commit
```

Display the BGP routing table of Device C.

```
[~DeviceC] display bgp routing-table
BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 9
Network          NextHop        MED     LocPrf  PrefVal Path/Ogn
*> 10.1.1.0/24    0.0.0.0      0          0      ?
*> 10.1.1.1/32   0.0.0.0      0          0      ?
*> 10.1.1.2/32   0.0.0.0      0          0      ?
*> 10.1.1.3/32   0.0.0.0      0          0      ?
*> 10.1.1.4/32   0.0.0.0      0          0      ?
*> 10.2.1.0/24   0.0.0.0      0          0      ?
*> 10.2.1.1/32   0.0.0.0      0          0      ?
*> 127.0.0.0      0.0.0.0      0          0      ?
*> 127.0.0.1/32  0.0.0.0      0          0      ?
```

Display the BGP routing table of Device D.

```
[~DeviceD] display bgp routing-table
BGP Local router ID is 4.4.4.4
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 3
```

Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
*>i 10.1.1.0/24	10.2.1.1	0	100	0	?
*>i 10.1.1.4/32	10.2.1.1	0	100	0	?
i 10.2.1.0/24	10.2.1.1	0	100	0	?

You can see that Device D has learned the IPv4 ARP Vlink direct route 10.1.1.4/32, whereas the route 10.1.1.3/32 has been filtered out.

- Step 6** Use the route-policy **policy2** to filter out the network segment route 10.1.1.0/24 on Device C when BGP routes are advertised.

Configure Device C.

```
[~DeviceC] ip ip-prefix prefix2 index 10 deny 10.1.1.0 24
[*DeviceC] ip ip-prefix prefix2 index 20 permit 0.0.0.0 0 less-equal 32
[*DeviceC] route-policy policy2 permit node 10
[*DeviceC-route-policy] if-match ip-prefix prefix2
[*DeviceC-route-policy] quit
[*DeviceC] bgp 100
[*DeviceC-bgp] peer 10.2.1.2 route-policy policy2 export
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
[~DeviceC] quit
<DeviceC> refresh bgp all export
```

Display the BGP routing table of Device D.

```
[~DeviceD] display bgp routing-table
BGP Local router ID is 4.4.4.4
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
Network          NextHop        MED     LocPrf    PrefVal Path/Ogn
*>i 10.1.1.4/32  10.2.1.1      0       100      0       ?
i 10.2.1.0/24   10.2.1.1      0       100      0       ?
```

You can find that the route 10.1.1.0/24 does not exist in the BGP routing table of Device D. As a result, Device D can communicate with Device B, but cannot communicate with Device A.

----End

Configuration Files

- Switch A configuration file

```
#
sysname switchA
#
vlan batch 10
#
interface Vlanif10
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet1/0/0
portswitch
undo shutdown
port link-type access
port default vlan 10
#
interface GigabitEthernet2/0/0
portswitch
undo shutdown
```

```
port link-type access
port default vlan 10
#
interface GigabitEthernet3/0/0
portswitch
undo shutdown
port link-type access
port default vlan 10
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.3 255.255.255.0
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.4 255.255.255.0
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
ip ip-prefix prefix1 index 10 permit 10.1.1.4 32
ip ip-prefix prefix2 index 10 deny 10.1.1.0 24
ip ip-prefix prefix2 index 20 permit 0.0.0.0 0 less-equal 32
#
route-policy policy1 permit node 10
if-match ip-prefix prefix1
#
route-policy policy2 permit node 10
if-match ip-prefix prefix2
#
arp vlink-direct-route advertise route-policy policy1
#
vlan batch 10
#
interface Vlanif10
ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet1/0/0
portswitch
undo shutdown
port link-type access
port default vlan 10
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0
#
bgp 100
router-id 3.3.3.3
peer 10.2.1.2 as-number 100
#
ipv4-family unicast
undo synchronization
import-route direct
peer 10.2.1.2 enable
```

```
peer 10.2.1.2 route-policy policy2 export
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0
#
#
bgp 100
router-id 4.4.4.4
peer 10.2.1.1 as-number 100
#
ipv4-family unicast
undo synchronization
peer 10.2.1.1 enable
#
return
```

Example for Importing IPv6 NDP Vlink Direct Routes to BGP4+

By importing IPv6 NDP Vlink direct routes to BGP4+, you can enable the remote device to obtain information about detailed routes in the VLAN, allowing precise control of data traffic.

Networking Requirements

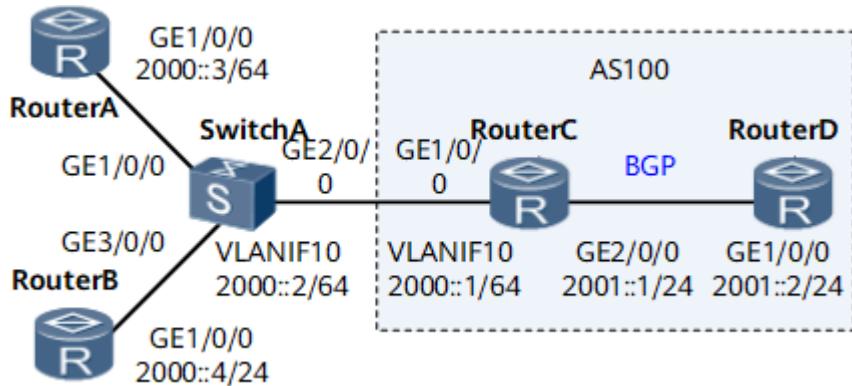
As networks develop, the VLAN technology is widely used. If a user outside a VLAN needs to communicate with users within the VLAN, advertising routes destined for the network segment of the VLAN can achieve this purpose. When users outside the VLAN need to know the IPv6 NDP Vlink direct routes of the VLAN, and apply different traffic policies to routes of the VLAN users, advertising the routes destined for the network segment of the VLAN cannot meet this requirement.

In this case, you can enable the function of IPv6 NDP Vlink direct route advertisement. On the network shown in [Figure 1-33](#), DeviceC is connected to two users through VLANIF interfaces. DeviceD needs to communicate with DeviceB, but not with DeviceA. In this case, you can enable the function of IPv6 NDP Vlink direct route advertisement on DeviceC, and use a route-policy to filter out subnet routes and the route to DeviceA.

Figure 1-33 Networking diagram of importing IPv6 NDP Vlink direct routes to BGP4+

NOTE

Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Create VLANIF interfaces on SwitchA and DeviceC and assign IPv6 addresses to the VLANIF interfaces, and ensure that DeviceA, DeviceB, SwitchA, and DeviceC can communicate with each other.
2. Enable BGP4+ on DeviceC and DeviceD, ensuring that DeviceC and DeviceD are able to advertise IPv6 routes to each other.
3. Enable the function of IPv6 NDP Vlink direct route advertisement on DeviceC.
4. Configure a route-policy on DeviceC, allowing IPv6 routes only from DeviceB to pass through.
5. Enable BGP4+ on DeviceC and import IPv6 direct routes to BGP4+, and use the route-policy to import IPv6 routes only from DeviceB to BGP4+.
6. Associate BGP4+ with the route-policy on DeviceC to filter out the network segment route of the VLAN so that DeviceD cannot learn the network segment route and can communicate with VLAN users based only on IPv6 NDP Vlink direct routes.

Data Preparation

To complete the configuration, you need the following data:

- ID of the VLAN in which SwitchA and DeviceC reside (the VLAN ID is 10 in this example)
- Router IDs and AS numbers of Devices C and D (router ID of DeviceC is 1.1.1.1 and router ID of DeviceD is 2.2.2.2, and Devices C and D are in AS 100 in this example)
- Route-policy used to filter direct routes (the route-policy is policy1 in this example)
- Route-policy used to advertise BGP4+ routes on DeviceC (the route-policy is policy2 in this example)

Procedure

Step 1 Configure an IP address for each interface.

Configure DeviceA.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
[*HUAWEI] commit
[~DeviceA] interface GigabitEthernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ipv6 enable
[*DeviceA-GigabitEthernet1/0/0] ipv6 address 2001:db8:2000::3 64
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

Configure DeviceB.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[*DeviceB] interface GigabitEthernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] undo shutdown
[*DeviceB-GigabitEthernet1/0/0] ipv6 enable
[*DeviceB-GigabitEthernet1/0/0] ipv6 address 2001:db8:2000::4 64
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

Configure DeviceC.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceC
[*HUAWEI] commit
[~DeviceC] interface GigabitEthernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ipv6 enable
[*DeviceC-GigabitEthernet2/0/0] ipv6 address 2001:db8:2001::1 64
[*DeviceC-GigabitEthernet2/0/0] commit
[~DeviceC-GigabitEthernet2/0/0] quit
```

Configure DeviceD.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceD
[*HUAWEI] commit
[~DeviceD] interface GigabitEthernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] ipv6 enable
[*DeviceD-GigabitEthernet1/0/0] ipv6 address 2001:db8:2001::2 64
[*DeviceD-GigabitEthernet1/0/0] commit
[~DeviceD-GigabitEthernet1/0/0] quit
```

Step 2 Configure basic VLAN functions. Create VLANIF 10 on SwitchA and DeviceC and assign IP addresses to the VLANIF interfaces.

Configure SwitchA.

```
<HUAWEI> system-view
[~HUAWEI] sysname SwitchA
[*HUAWEI] commit
[~SwitchA] vlan 10
[*SwitchA-vlan10] quit
[*SwitchA] interface GigabitEthernet 1/0/0
[*SwitchA-GigabitEthernet1/0/0] portswitch
[*SwitchA-GigabitEthernet1/0/0] port link-type access
[*SwitchA-GigabitEthernet1/0/0] port default vlan 10
[*SwitchA-GigabitEthernet1/0/0] quit
[*SwitchA] interface GigabitEthernet 2/0/0
[*SwitchA-GigabitEthernet2/0/0] portswitch
[*SwitchA-GigabitEthernet2/0/0] port link-type access
[*SwitchA-GigabitEthernet2/0/0] port default vlan 10
[*SwitchA-GigabitEthernet2/0/0] quit
```

```
[*SwitchA] interface GigabitEthernet 3/0/0
[*SwitchA-GigabitEthernet3/0/0] portswitch
[*SwitchA-GigabitEthernet3/0/0] port link-type access
[*SwitchA-GigabitEthernet3/0/0] port default vlan 10
[*SwitchA-GigabitEthernet3/0/0] quit
[*SwitchA] interface Vlanif 10
[*SwitchA-Vlanif10] ipv6 enable
[*SwitchA-Vlanif10] ipv6 address 2001:db8:2000::2 64
[*SwitchA-Vlanif10] commit
[~SwitchA-Vlanif10] quit
```

Configure DeviceC.

```
[~DeviceC] vlan 10
[~DeviceC-vlan10] quit
[*DeviceC] interface GigabitEthernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] portswitch
[*DeviceC-GigabitEthernet1/0/0] port link-type access
[*DeviceC-GigabitEthernet1/0/0] port default vlan 10
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface Vlanif 10
[*DeviceC-Vlanif10] ipv6 enable
[*DeviceC-Vlanif10] ipv6 address 2001:db8:2000::1 64
[*DeviceC-Vlanif10] commit
[~DeviceC-Vlanif10] quit
```

Step 3 Configure BGP4+ between DeviceC and DeviceD.

Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] router-id 1.1.1.1
[*DeviceC-bgp] peer 2001:db8:2001::2 as-number 100
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:2001::2 enable
[*DeviceC-bgp-af-ipv6] commit
[~DeviceC-bgp-af-ipv6] quit
[~DeviceC-bgp] quit
```

Configure DeviceD.

```
[~DeviceD] bgp 100
[*DeviceD-bgp] router-id 2.2.2.2
[*DeviceD-bgp] peer 2001:db8:2001::1 as-number 100
[*DeviceD-bgp] ipv6-family unicast
[*DeviceD-bgp-af-ipv6] peer 2001:db8:2001::1 enable
[*DeviceD-bgp-af-ipv6] commit
[~DeviceD-bgp-af-ipv6] quit
[~DeviceD-bgp] quit
```

After the configuration is complete, run the **display bgp ipv6 peer** command to view status of BGP4+ peer relationships. The command output shows that the IBGP peer relationship has been established between DeviceC and DeviceD. Use the display on DeviceD as an example.

```
[~DeviceD] display bgp ipv6 peer
BGP local router ID : 2.2.2.2
Local AS number : 100
Total number of peers : 1          Peers in established state : 1
Peer      V      AS  MsgRcvd  MsgSent  OutQ  Up/Down    State PrefRcv
2001:db8:2001::1      4      100     64     59    0 00:52:15 Established    0
```

Step 4 Configure BGP4+ on DeviceC and import direct routes to BGP4+. Then view the routing tables of Devices C and D.

Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] import-route direct
[*DeviceC-bgp-af-ipv6] commit
[~DeviceC-bgp-af-ipv6] quit
[~DeviceC-bgp] quit
```

Display the BGP4+ routing table of DeviceC.

```
[~DeviceC] display bgp ipv6 routing-table
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 12
*> Network ::1                           PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :2001:db8:2000::             PrefixLen : 64
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :2001:db8:2000::1            PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :2001:db8:2000::2            PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :2001:db8:2001::            PrefixLen : 64
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :2001:db8:2001::1           PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :FE80::                      PrefixLen : 10
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :FE80::2E0:39FF:FE18:8300     PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :FE80::2E0:91FF:FE4F:8100     PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
*> Network :FE80::2E0:9BFF:FE7E:7800     PrefixLen : 128
    NextHop ::                           LocPrf   :
    MED      :0                         PrefVal  :0
    Label    :
    Path/Ogn :?
```

Display the BGP4+ routing table of DeviceD.

```
[~DeviceD] display bgp ipv6 routing-table
BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
*>i Network : 2001:db8:2000:: PrefixLen : 64
    NextHop : 2001:db8:2001::1 LocPrf : 100
    MED : 0 PrefVal : 0
    Label :
    Path/Ogn : ?
i Network : 2001:db8:2001:: PrefixLen : 64
    NextHop : 2001:db8:2001::1 LocPrf : 100
    MED : 0 PrefVal : 0
    Label :
    Path/Ogn : ?
```

You can see that DeviceD has not learned the two IPv6 NDP Vlink direct routes 2001:db8:2000::3/128 and 2001:db8:2000::4/128.

- Step 5** Enable the function of IPv6 NDP Vlink direct route advertisement on DeviceC and configure the route-policy **policy1** to filter out the routes to the network segment of the VLAN and the IPv6 NDP Vlink direct route from DeviceA, 2001:db8:2000::3/128.

Configure DeviceC.

```
[~DeviceC] ip ipv6-prefix prefix1 permit 2001:db8:2000::4 128
[*DeviceC] route-policy policy1 permit node 10
[*DeviceC-route-policy] if-match ipv6 address prefix-list prefix1
[*DeviceC-route-policy] quit
[*DeviceC] ipv6 nd vlink-direct-route advertise route-policy policy1
[*DeviceC] commit
```

Display the BGP4+ routing table of DeviceC.

```
[~DeviceC] display bgp ipv6 routing-table
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 12
*> Network ::1 PrefixLen : 128
    NextHop :: LocPrf :
    MED : 0 PrefVal : 0
    Label :
    Path/Ogn : ?
*> Network : 2001:db8:2000:: PrefixLen : 64
    NextHop :: LocPrf :
    MED : 0 PrefVal : 0
    Label :
    Path/Ogn : ?
*> Network : 2001:db8:2000::1 PrefixLen : 128
    NextHop :: LocPrf :
    MED : 0 PrefVal : 0
    Label :
    Path/Ogn : ?
*> Network : 2001:db8:2000::2 PrefixLen : 128
    NextHop :: LocPrf :
    MED : 0 PrefVal : 0
    Label :
    Path/Ogn : ?
*> Network : 2001:db8:2000::3 PrefixLen : 128
    NextHop :: LocPrf :
```

```
MED    : 0          PrefVal : 0
Label   :
Path/Ogn : ?
*> Network : 2001:db8:2000::4      PrefixLen : 128
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*> Network : 2001:db8:2001::      PrefixLen : 64
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*> Network : 2001:db8:2001::1     PrefixLen : 128
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*> Network : FE80::          PrefixLen : 10
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*> Network : FE80::2E0:39FF:FE18:8300  PrefixLen : 128
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*> Network : FE80::2E0:91FF:FE4F:8100  PrefixLen : 128
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*> Network : FE80::2E0:9BFF:FE7E:7800  PrefixLen : 128
  NextHop : ::           LocPrf   :
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
```

Display the BGP4+ routing table of DeviceD.

```
[~DeviceD] display bgp ipv6 routing-table
BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 3
*>i Network : 2001:db8:2000::      PrefixLen : 64
  NextHop : 2001:db8:2001::1      LocPrf   : 100
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
*>i Network : 2001:db8:2000::4      PrefixLen : 128
  NextHop : 2001:db8:2001::1      LocPrf   : 100
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
i Network : 2001:db8:2001::      PrefixLen : 64
  NextHop : 2001:db8:2001::1      LocPrf   : 100
  MED     : 0           PrefVal  : 0
  Label   :
  Path/Ogn : ?
```

You can see that DeviceD has learned the IPv6 NDP Vlink direct route 2001:db8:2000::4/128, whereas the route 2001:db8:2000::3/128 has been filtered out.

Step 6 Use the route-policy **policy2** to filter out the network segment route 2001:db8:2000::/64 on DeviceC when BGP4+ routes are advertised.

Configure DeviceC.

```
[~DeviceC] ip ipv6-prefix prefix2 index 10 deny 2001:db8:2000:: 64
[*DeviceC] ip ipv6-prefix prefix2 index 20 permit :: 0 less-equal 128
[*DeviceC] route-policy policy2 permit node 10
[*DeviceC-route-policy] if-match ipv6 address prefix-list prefix2
[*DeviceC-route-policy] quit
[*DeviceC] bgp 100
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:2001::2 route-policy policy2 export
[*DeviceC-bgp-af-ipv6] commit
[~DeviceC-bgp-af-ipv6] quit
[~DeviceC-bgp] quit
[~DeviceC] quit
<DeviceC> refresh bgp all export
```

Display the BGP4+ routing table of DeviceD.

```
[~DeviceD] display bgp ipv6 routing-table
BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
*>i Network : 2001:db8:2000::4                  PrefixLen : 128
    NextHop : 2001:db8:2001::1                  LocPrf   : 100
    MED     : 0                                  PrefVal  : 0
    Label   :
    Path/Ogn : ?
i Network : 2001:db8:2001::                  PrefixLen : 64
    NextHop : 2001:db8:2001::1                  LocPrf   : 100
    MED     : 0                                  PrefVal  : 0
    Label   :
    Path/Ogn : ?
```

You can see that the route 2001:db8:2000::/64 does not exist in the BGP4+ routing table of DeviceD. As a result, DeviceD can communicate with DeviceB, but cannot communicate with DeviceA.

----End

Configuration Files

- SwitchA configuration file

```
# 
sysname SwitchA
#
vlan batch 10
#
interface Vlanif10
    ipv6 enable
    ipv6 address 2001:db8:2000::2/64
#
interface GigabitEthernet1/0/0
    portswitch
    undo shutdown
    port link-type access
    port default vlan 10
#
interface GigabitEthernet2/0/0
    portswitch
    undo shutdown
```

```
port link-type access
port default vlan 10
#
interface GigabitEthernet3/0/0
portswitch
undo shutdown
port link-type access
port default vlan 10
#
return
```

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2000::3/64
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2000::4/64
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
ip ipv6-prefix prefix1 index 10 permit 2001:db8:2000::4 128
ip ipv6-prefix prefix2 index 10 deny 2001:db8:2000:: 64
ip ipv6-prefix prefix2 index 20 permit :: 0 less-equal 128
#
route-policy policy1 permit node 10
if-match ip-prefix prefix1
if-match ipv6 address prefix-list prefix1
#
route-policy policy2 permit node 10
if-match ipv6 address prefix-list prefix2
#
ipv6 nd vlink-direct-route advertise route-policy policy1
#
vlan batch 10
#
interface Vlanif10
ipv6 enable
ipv6 address 2001:db8:2000::1/64
#
interface GigabitEthernet1/0/0
portswitch
undo shutdown
port link-type access
port default vlan 10
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2001::1/64
#
bgp 100
router-id 1.1.1.1
peer 2001:db8:2001::2 as-number 100
```

```
#  
ipv4-family unicast  
undo synchronization  
#  
ipv6-family unicast  
undo synchronization  
import-route direct  
peer 2001:db8:2001::2 enable  
peer 2001:db8:2001::2 route-policy policy2 export  
#  
return
```

- DeviceD configuration file

```
#  
sysname DeviceD  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:2001::2/64  
#  
bgp 100  
router-id 2.2.2.2  
peer 2001:db8:2001::1 as-number 100  
#  
ipv4-family unicast  
undo synchronization  
#  
ipv6-family unicast  
undo synchronization  
peer 2001:db8:2001::1 enable  
#  
return
```

1.1.2 IPv4 Static Route Configuration

1.1.2.1 Static Routes Description

1.1.2.1.1 Overview of Static Routes

Definition

Static routes are special routes that are configured by network administrators.

Purpose

On a simple network, only static routes can ensure that the network runs properly. If a router cannot run dynamic routing protocols or cannot generate routes to a destination network, you can configure static routes on the router.

Route selection can be controlled using static routes. Properly configuring and using static routes can improve network performance and guarantee the required bandwidth for important applications. When a network fault occurs or the network topology changes, however, static routes must be changed manually by the administrator.

1.1.2.1.2 Understanding Static Routes

Components

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, you can run the **ip route-static** command to configure a static route, which consists of the following components:

- **Destination address and mask**
- **Outbound interface and next hop address**

Destination Address and Mask

The IPv4 destination address in a static route is expressed in dotted decimal notation, while a mask can be expressed either in dotted decimal or CIDR notation.

The IPv6 destination address in a static route is a 32-digit hexadecimal number, while a prefix is expressed in CIDR notation and ranges from 0 to 128.

Outbound Interface and Next Hop Address

When creating a static route, you can specify *interface-type interface-number*, *nexthop-address*, or both. In addition, you can configure the Next-Table function, that is, only a VPN instance name (public in the case of the public network) is specified as the next hop of a static route, and no outbound interface or next hop address is specified. You can configure the parameters as required.

Every route requires a next-hop address. Before sending a packet, a device needs to search its routing table for the route matching the destination address in the packet using the longest match rule. The link layer can find the corresponding link-layer address and then forward the packet only when a next-hop IP address is available.

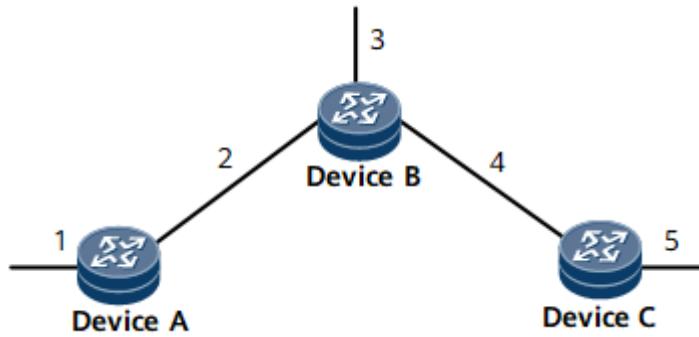
When specifying an outbound interface, note the following:

- For a Point-to-Point (P2P) interface, if an outbound interface is specified, the next hop address is the address of the remote interface connected to the outbound interface. For example, when a GE interface is configured with PPP encapsulation and obtains the remote IP address through PPP negotiation, you can specify only an outbound interface, without the need to specify a next hop address.
- Non-Broadcast Multiple-Access (NBMA) interfaces are applicable to Point-to-Multipoint networks. Therefore, IP routes and the mappings between IP addresses and link layer addresses are required. In this case, you need to configure next hop addresses.
- When configuring static routes, you are advised not to specify a broadcast interface (such as an Ethernet interface) or a virtual template (VT) interface as the outbound interface. Ethernet interfaces are broadcast interfaces, and each VT interface can be associated with multiple virtual access interfaces. If either of the two types of interfaces is specified as the outbound interface, multiple next hops exist and the next hop cannot be determined. In actual applications, to specify a broadcast interface (such as an Ethernet interface) or a VT interface as the outbound interface, you need to specify a next hop address along with the outbound interface.

Application Scenarios for Static Routes

In **Figure 1-34**, the network topology is simple, and network communication can be implemented through static routes. You need to specify an address for each physical network, identify indirectly connected physical networks for each device, and configure static routes for indirectly connected physical networks.

Figure 1-34 Networking for static routes



In this example, static routes to networks 3, 4, and 5 need to be configured on Device A; static routes to networks 1 and 5 need to be configured on Device B; static routes to networks 1, 2, and 3 need to be configured on Device C.

Default Static Route

Default routes are a special kind of routes, and default static routes are manually configured. The default route is used when no matched entry is available in the routing table. In an IPv4 routing table, the destination address and subnet mask of a default route are both 0.0.0.0. In an IPv6 routing table, the destination address and prefix of a default route are both ::.

If the destination address of a packet does not match any entry in the routing table, the device selects the default route to forward this packet. If no default route exists and the destination address of the packet does not match any entry in the routing table, the packet is discarded. An Internet Control Message Protocol (ICMP) packet is then sent, informing the originating host that the destination host or network is unreachable.

The static route with the destination address and mask 0s (0.0.0.0 0.0.0.0) configured using the **ip route-static** command is a default route intended to simplify network configuration.

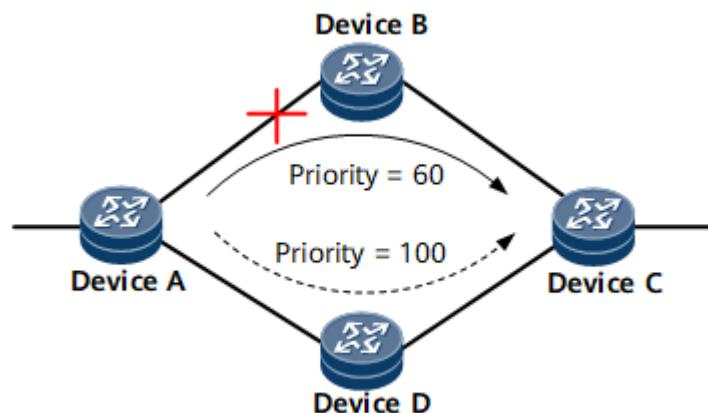
In [Figure 1-34](#), because the next hop of the packets from Device A to networks 3, 4, and 5 is Device B, a default route can be configured on Device A to replace the three static routes destined for networks 3, 4, and 5. Similarly, only a default route from Device C to Device B needs to be configured to replace the three static routes destined for networks 1, 2, and 3.

Floating Static Routes

Different static routes can be configured with different priorities so that routing management policies can be flexibly applied. Route backup can be implemented by specifying different priorities for multiple routes to the same destination.

In **Figure 1-35**, there are two static routes from Device A to Device C. In most cases, the only Active route is the static route with Device B as the next hop in the routing table because it has a higher priority. The other static route with Device D as the next hop functions as a backup route. The backup route is only activated to forward traffic if the primary link fails. After the primary link recovers, the static route with Device B as the next hop becomes Active to take over the traffic. Therefore, the backup route is also called a floating static route. The floating static route becomes ineffective if a fault occurs on the link between Device B and Device C.

Figure 1-35 Networking for a floating static route

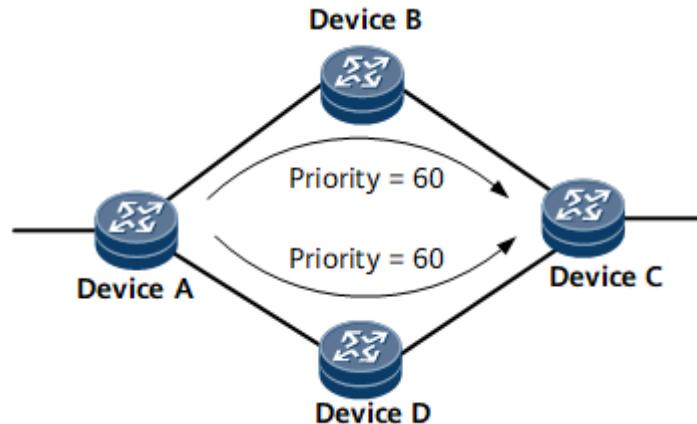


Load Balancing Among Static Routes

Routes to the same destination with the same priority can be used to load-balance traffic.

As shown in **Figure 1-36**, there are two static routes with the same priority from Device A to Device C. The two routes both exist in the routing table and forward traffic at the same time.

Figure 1-36 Load balancing among static routes



FRR for Static Routes

When routes are delivered to the routing management (RM) module, the optimal route is delivered with a backup route. If the optimal route fails, traffic is immediately switched to the backup route, minimizing traffic loss.

You need to configure two routes with the same prefix but different priorities to implement FRR. The route with the higher priority is the primary route, and the route with the lower priority is the backup route. FRR is implemented only on static routes that are manually configured. That is, FRR is not implemented on recursive next hops.

Functions

IPv4 Static Routes

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports common static routes and the static routes associated with VPN instances. The static routes associated with VPN instances are used to manage VPN routes. For details about VPN instances, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description - VPN*.

Attributes and Functions of IPv6 Static Routes

Similar to IPv4 static routes, IPv6 static routes are configured by the administrator and are applicable to simple IPv6 networks.

The major difference between IPv6 static routes and IPv4 static routes lies in their destination addresses and next hop addresses.

An IPv6 static route with destination address ::/0 (mask length 0) is a default IPv6 route. If the destination address of an IPv6 packet fails to match any entry in the routing table, a router selects the default IPv6 route to forward the IPv6 packet.

BFD for Static Route

Different from dynamic routing protocols, static routes do not have a detection mechanism. If a fault occurs on a network, an administrator must manually

address it. Bidirectional Forwarding Detection (BFD) for static route is introduced to associate a static route with a BFD session so that the BFD session can detect the status of the link that the static route passes through.

After BFD for static route is configured, each static route can be associated with a BFD session. In addition to route selection rules, whether a static route can be selected as the optimal route is subject to BFD session status.

- If a BFD session associated with a static route detects a link failure when the BFD session is Down, the BFD session reports the link failure to the system. The system then deletes the static route from the IP routing table.
- If a BFD session associated with a static route detects that a faulty link recovers when the BFD session is Up, the BFD session reports the fault recovery to the system. The system then adds the static route to the IP routing table again.
- By default, a static route can still be selected even though the BFD session associated with it is AdminDown (triggered by the **shutdown** command run either locally or remotely). However, if the system is restarted, the BFD session needs to be re-negotiated. Whether the static route can participate in route selection depends on the re-negotiated BFD status.

BFD for static route works in single-hop detection or multi-hop detection mode.

- Single-hop detection

In single-hop detection mode, the configured outbound interface and next hop address are the information about the directly connected next hop. The outbound interface associated with the BFD session is the outbound interface of the static route, and the peer address is the next hop address of the static route.

- Multi-hop detection

In multi-hop detection mode, only the next hop address is configured. Therefore, the static route must recurse to the directly connected next hop and outbound interface. The peer address of the BFD session is the original next hop address of the static route, and the outbound interface is not specified. In most cases, the original next hop is an indirect next hop. Multi-hop detection is performed on the static routes that support route recursion.

NOTE

For details about BFD, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series RouterFeature Description - Network Reliability*.

NQA for Static Route

Background

Static routes do not have a dedicated detection mechanism. If a link fails, the corresponding static route will not be automatically deleted from the IP routing table. In this case, intervention of a network administrator is required. This delays the link switchover and may cause lengthy service interruptions.

BFD for static route can use BFD sessions to monitor the link status of a static route. However, BFD for static route requires that both ends of a link support BFD. As a result, it cannot be implemented in some scenarios. NQA for static route can solve this problem.

Table 1-9 compares BFD for static route and NQA for static route.

Table 1-9 Comparison between BFD for static route and NQA for static route

Item	BFD for Static Route	NQA for Static Route
Detection mode	Bidirectional session	Unidirectional detection
Requirements for devices	Both ends must support BFD.	NQA is required on only one end.
Detection speed	Millisecond-level	Second-level

Related Concepts

NQA helps carriers monitor network quality of service (QoS) in real time, and can be used to diagnose the fault if a network fails.

NQA relies on a test instance to monitor the link status. The two ends of an NQA test are called the NQA client and the NQA server. An NQA test is initiated by the NQA client. NQA test results are classified into the following types:

- Success: The test is successful. It instructs the routing management module to set the status of the static route to active and add the static route to the routing table.
- Failed: The test fails. It instructs the routing management module to set the status of the static route to inactive and delete the static route from the routing table.
- No result: The test is running and no result has been obtained. If the test result is no result, the status of the static route is not changed.



For NQA details, see "System Monitor" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description*.

Implementation

NQA for static route associates an NQA test instance with a static route and uses the NQA test instance to monitor the link status. The routing management module determines whether a static route is active based on the NQA test result. If the static route is inactive, the routing management module deletes it from the IP routing table and selects a normal backup link for data forwarding, which prevents lengthy service interruptions.

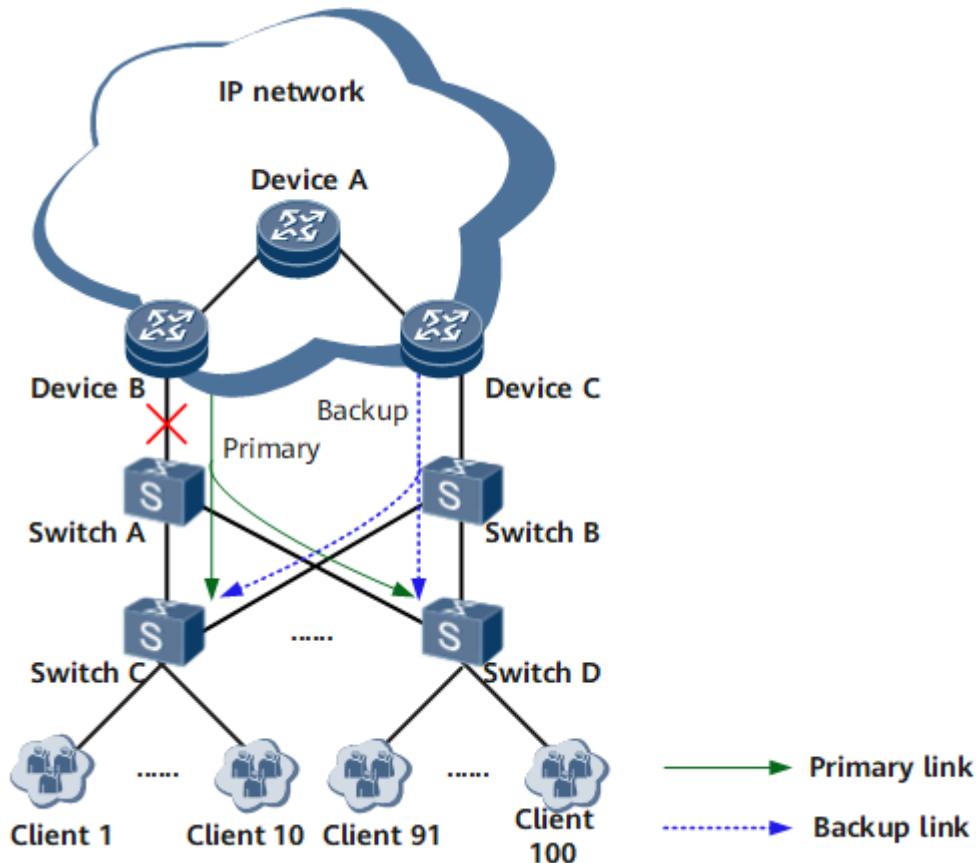
In **Figure 1-37**, each access switch is connected to 10 clients, and a total of 100 clients exist. Because no dynamic routing protocol can be used between DeviceB and clients, static routes to the clients need to be configured on DeviceB. To ensure network stability, the same configuration is performed on DeviceC for backup.

DeviceA, DeviceB, and DeviceC run a dynamic routing protocol and can learn routes from each other. On DeviceB and DeviceC, the dynamic routing protocol is configured to import static routes, and different costs are configured for the static

routes. In this way, DeviceA can learn the routes to clients from DeviceB and DeviceC through the dynamic routing protocol. DeviceA then determines the primary and backup links based on the costs.

NQA for static route is configured on DeviceB, and an NQA test instance is used to monitor the status of the primary link. If the primary link fails, the corresponding static route is deleted and downlink traffic switches to the backup link. When both the links are running properly, downlink traffic is preferentially transmitted along the primary link.

Figure 1-37 Network diagram of NQA for static route



NOTE

NQA test instances can monitor the links of IPv4 and IPv6 static routes. The mechanisms for monitoring IPv4 and IPv6 static routes are the same.

Each static route can be associated with only one NQA test instance.

Usage Scenario

NQA for static route applies to a network where BFD for static route cannot be deployed due to device limitations. For example, user devices access the network through the switch, OLT, DSLAM, MSAN, or xDSL mode.

Benefits

It can rapidly and periodically detect the link status of static routes and implement rapid primary/backup link switchovers, preventing lengthy service interruptions.

NQA Group for Static Route

Background

Static routes do not have a dedicated detection mechanism. If a link fails, the corresponding static route will not be automatically deleted from the IP routing table. In this case, intervention of a network administrator is required. This delays the link switchover and may cause lengthy service interruptions.

NQA for static route can monitor only a single link. To enable a static route to use NQA test instances to monitor multiple links, you can bind these NQA test instances to the same NQA group and associate the static route with the NQA group.

Related Concepts

NQA helps carriers monitor network quality of service (QoS) in real time, and can be used to diagnose and locate the fault if a network fails.

NQA relies on a test instance to monitor the link status. The two ends of an NQA test are called the NQA client and the NQA server. An NQA test is initiated by the NQA client. NQA test results are classified into the following types:

- Success: The test is successful. It instructs the routing management module to set the status of the static route to active and add the static route to the routing table.
- Failed: The test fails. It instructs the routing management module to set the status of the static route to inactive and delete the static route from the routing table.
- No result: The test is running and no result has been obtained. If the test result is no result, the status of the static route is not changed.

NOTE

For NQA details, see "System Monitor" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description*.

Implementation

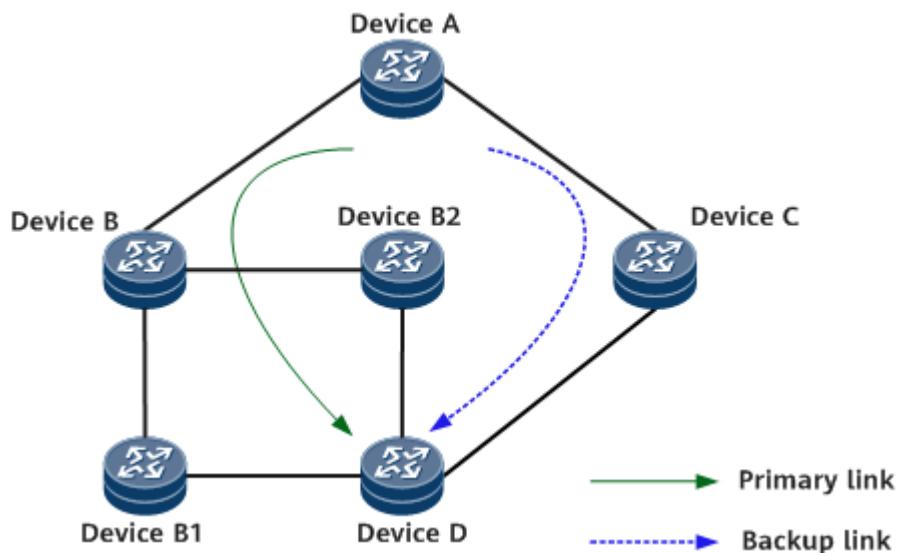
NQA group for static route monitors the link status of a static route through association between the static route and an NQA group that consists of multiple NQA test instances. The RM module determines whether the static route is active based on the test result of the NQA group. If the static route is inactive, the RM module deletes it from the IP routing table and selects an available backup link for data forwarding, which prevents lengthy service interruptions.

On the network shown in [Figure 1-38](#), default static routes are configured on Device B and Device C to divert traffic. To ensure network stability, the link between Device A and Device C work in backup mode.

Device B and Device D are connected through two links by way of Device B1 and Device B2. Device A, Device B, and Device C run BGP and can learn routes from each other. Device B and Device C are configured to import static routes into BGP and set different PrefVal values so that Device A can use BGP to learn user routes from Device B and Device C. Device A then determines the primary and backup links based on the PrefVal values.

NQA group for static route is configured on Device B, and two NQA test instances are bound to the NQA group to monitor the status of the two links from Device B to Device D. The status of the NQA group reflects the status of the primary link. If the primary link fails, advertisement of the static route is withdrawn so that user traffic is transmitted through the backup link. When both the links are running properly, user traffic is preferentially transmitted along the primary link.

Figure 1-38 Network diagram of NQA group for static route



NOTE

NQA groups can monitor the links of IPv4 and IPv6 static routes. The mechanisms for monitoring IPv4 and IPv6 static routes are the same.

Each static route can be associated with only one NQA group.

Usage Scenario

NQA group for static route applies to the scenario where whether a static route is active is determined by using multiple NQA test instances to monitor the status of multiple links.

Benefits

It can rapidly and periodically detect the link status of static routes and implement rapid primary/backup link switchovers, preventing lengthy service interruptions.

Static Route Permanent Advertisement

Background

When the link over which a static route runs fails, the static route will be deleted from the IP routing table to trigger a route re-selection. After a new route is selected, traffic is switched to the new route. Some carriers, however, may require that specific traffic always travel along a fixed link, regardless of the link status. Static route permanent advertisement is introduced to meet this service need.

Implementation

With static route permanent advertisement, a static route can still be advertised and added to the IP routing table for route selection even when the link over which the static route runs fails. After static route permanent advertisement is configured, the static route can be advertised and added to the IP routing table in both of the following scenarios:

- An outbound interface is configured for the static route, and the outbound interface has an IP address. Static route permanent advertisement is not affected no matter whether the outbound interface is Up.
- No outbound interface is configured for the static route. Static route permanent advertisement is not affected no matter whether the static route can obtain an outbound interface through route recursion.

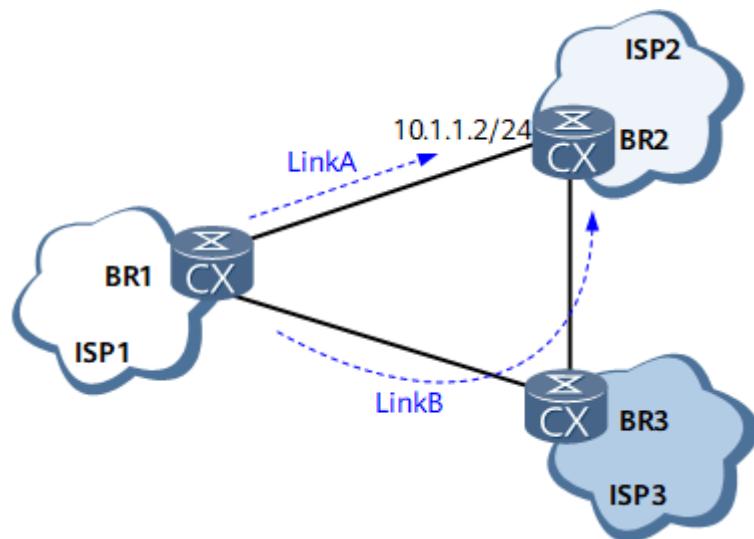
NOTE

After static route permanent advertisement is enabled, a static route always remains in the IP routing table regardless of route reachability. If the destination of the route becomes unreachable, traffic interruption occurs.

Typical Networking

On the network shown in [Figure 1-39](#), BR1, BR2, and BR3 belong to ISP1, ISP2, and ISP3 respectively. Two links (Link A and Link B) exist between BR1 and BR2, but ISP1 expects its service traffic destined for ISP2 to be always transmitted over Link A.

Figure 1-39 Networking with static route permanent advertisement



A direct EBGP peer relationship is established between BR1 and BR2. A static route is created on BR1, with 10.1.1.2/24 (IP address of BR2) as the destination address and the local interface connected to BR2 as the outbound interface.

Without static route permanent advertisement, Link A is used to transmit traffic. If Link A fails, BGP will switch the traffic to Link B.

With static route permanent advertisement, Link A is used to transmit traffic regardless of whether the destination is reachable through Link A. If Link A fails, no link switchover is performed, causing traffic interruption. To check whether the destination is reachable through the static route, ping the destination address of the static route to which static route permanent advertisement is applied.

Association Between LDP and Static Routes

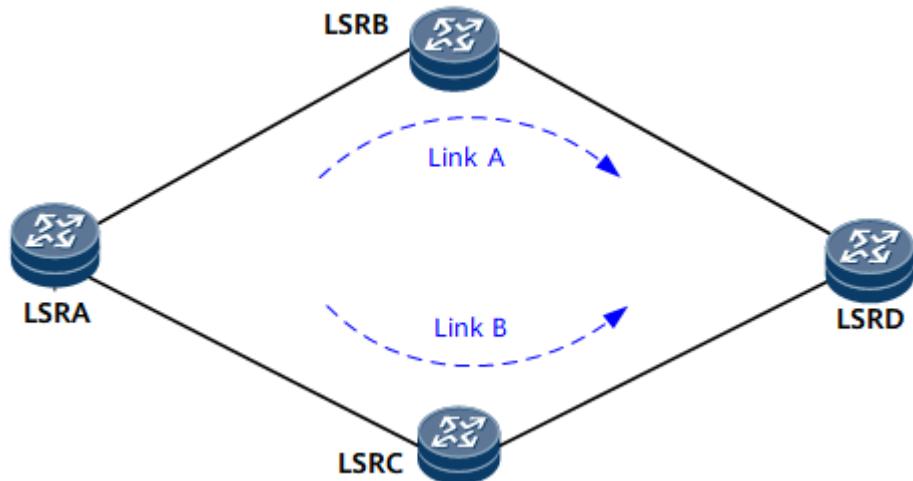
Background

On a network with a backup path between label switching routers (LSRs), packet loss may occur during a traffic switchover or switchback because the status of a static route is different from that of a Label Distribution Protocol (LDP) session. To resolve this problem, configure association between LDP and static routes.

Typical Networking

Figure 1-40 shows the typical networking of association between LDP and static routes. LSRA and LSRD interwork through static routes. Primary and backup static routes are deployed on LSRA, with the next-hop devices being LSRB and LSRC, respectively. Primary and backup LDP LSPs are established based on the static routes. The primary LSP uses Link A, and the backup LSP uses Link B. In normal cases, Link A is preferred. Association between LDP and static routes in switchover and switchback scenarios is described as follows.

Figure 1-40 Networking of LSP switching scenario where association between LDP and static routes is configured



Switchover scenario

In the switchover scenario, traffic of the primary static route is not switched to the backup link when the LDP session on the primary link fails (not because of a link fault). As a result, traffic on the LSP over the primary link is interrupted.

After an LDP session is established, LSP traffic travels along the primary link, Link A (LSRA → LSRB → LSRD). If the LDP session between LSRA and LSRB is interrupted, traffic of the primary LSP is switched immediately to the backup link, Link B (LSRA → LSRC → LSRD). However, because the link between LSRA and LSRB is normal, traffic of the primary static route is not switched to the backup link. The asynchronous state between LDP and the primary static route causes an LSP traffic interruption.

If association between LDP and static routes is enabled, traffic is automatically switched to the backup link when the LDP session goes Down, ensuring uninterrupted traffic forwarding.

Switchback scenario

In the switchback scenario, when the primary link recovers from a fault, the traffic of the primary static route is switched back to Link A earlier than the traffic of the primary LSP because the convergence of static routes is faster than that of LDP LSPs. As a result, the backup LSP on Link B cannot be used, and the LSP on Link A has not been set up yet. As a result, LSP traffic is interrupted.

If the link between LSRA and LSRB fails, traffic is switched immediately to the backup link, Link B (LSRA → LSRC → LSRD). After the link between LSRA and LSRB recovers, traffic of the primary static route is immediately switched back to Link A (LSRA → LSRB → LSRD). However, the backup LSP cannot be used, and the LSP on Link A has not recovered yet. As a result, traffic is interrupted.

If association between LDP and static routes is enabled, the static route on Link A becomes active only when the LDP session on Link A goes Up. In this manner, the states of the primary static route and LSP are asynchronous during the switchback, which prevents traffic loss.

Usage Scenario

Association between LDP and static routes applies to scenarios where a static route backup path exists between LSRs.

Benefits

Association between LDP and static routes ensures state consistency between LDP and static routes, prevents traffic loss, and improves network reliability.

1.1.2.2 IPv4 Static Route Configuration

Static routes are applicable to networks with simple structures. Properly configuring and using static routes can improve network performance and guarantee the required bandwidth for important applications.

1.1.2.2.1 Overview of IPv4 Static Routes

Configuring IPv4 static routes can implement the interworking of simple networks.

Definition

Static routes are special routes that are configured by network administrators.

Purpose

On a simple network, only static routes can ensure that the network runs properly. If a router cannot run dynamic routing protocols or cannot generate routes to a destination network, you can configure static routes on the router.

Route selection can be controlled using static routes. Properly configuring and using static routes can improve network performance and guarantee the required bandwidth for important applications. When a network fault occurs or the network topology changes, however, static routes must be changed manually by the administrator.

1.1.2.2 Configuration Precautions for IPv4 static route

Feature Requirements

Table 1-10 Feature requirements

Feature Requirements	Series	Models
Static routes cannot be recursed to SRv6 TE Policy Color-Only tunnels. If the route (such as a BGP route) on which the next hop of a static route is recursed uses a Color-Only tunnel, the static route inherits the Color-Only tunnel as the next hop. However, BFD for locator fast switching is not supported.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
The static routes to participate in FRR must have different priorities configured.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
When the number of configured static routes for load balancing exceeds the product specification limit, the preferred static route may change after the active preferred route is modified or the active/standby switchover is performed twice.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
Before you associate a static route with a BFD session, the session must have been created.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
If recursion depth-based static route selection is configured, the number of static routes for load balancing may decrease. As a result, static routes may fail to implement load balancing.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X

Feature Requirements	Series	Models
<p>When static routes are deployed, the following features are mutually exclusive:</p> <ul style="list-style-type: none">1. BFD (Bidirectional Forwarding Detection) for static route2. NQA (Network Quality Analysis) for static route3. Association between static routes and EFM44. Permanent advertisement of static routes <p>Determine which feature to use based on the actual network conditions:</p> <ul style="list-style-type: none">■ BFD for static routes enables a static route to be bound to a BFD session. The BFD session detects the link status of the static route and determines whether to activate the static route.■ Deploying NQA for static routes uses NQA test instances to detect the status of the link where the static route resides and determines whether to activate the static route based on the NQA test results. NQA for static routes can be deployed in special scenarios, for example, when BFD is not applicable to links between different ISPs or a link segment does not support BFD.■ After EFM is associated with static routes, the system responds to the EFM Up/Down event of a specified interface and determines whether to activate static routes. This controls route advertisement and directs remote traffic.■ Permanent advertisement of static routes can be deployed if the customer wants to determine the forwarding path of service traffic so that traffic is not switched to another path even if a link fault occurs.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X

Feature Requirements	Series	Models
The route recursive-lookup tunnel command can be used to recurse IPv4 VPN static routes whose next hops are on the public network to tunnels on the public network. After this command is run, the system preferentially recurses a static route to an IP route. If no matching IP route is available for recursion, the system does not recurse the static route to a tunnel. If a matching BGP route is available for recursion, the static route recurses to the BGP route. If a matching non-BGP route is available for recursion, the static route recurses to a tunnel based on the tunnel policy. If a matching network segment (non-host route) is available for recursion, the network segment address is used to recurse the static route to a tunnel. Therefore, the static route can recurse to a public network tunnel based on the next hop address only if a matching non-BGP route corresponding to the next hop is available.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

1.1.2.2.3 Configuring IPv4 Static Routes

On a network, you can control route selection by configuring IPv4 Static Routes.

Usage Scenario

IPv4 static routes can be configured for a network with simple structure to achieve connectivity.

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports common static routes and those associated with VPN instances. The static routes associated with VPN instances are used to manage VPN routes. For details about VPN instances, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - VPN*.

Pre-configuration Tasks

Before configuring IPv4 Static Routes, configure parameters of the link layer protocol and IP addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.

Creating IPv4 Static Routes

To create an IPv4 static route, configure its destination address, outbound interface, and next hop.

Context

When creating an IPv4 static route, you need the following information:

- Destination address and mask

In the **ip route-static** command, the IPv4 address is expressed in dotted decimal notation, and the mask is expressed either in dotted decimal notation or represented by the mask length.

- Outbound interface and next hop address

When creating a static route, you can specify *interface-type interface-name*, *nexthop-address*, or both. In addition, you can configure the Next-Table function, that is, only a VPN instance name (public in the case of the public network) is specified as the next hop of a static route, and no outbound interface or next hop address is specified. You can configure the parameters as required.

Actually, each routing entry requires a next hop address. Before sending a packet, a device needs to search its routing table for the route matching the destination address in the packet based on the longest match rule. The device can find the associated link layer address to forward the packet only when the next hop address of the packet is available.

When specifying an outbound interface, note the following rules:

- For a Point-to-Point (P2P) interface, if the outbound interface is specified, the next hop address is the address of the remote interface connected to the outbound interface. For example, when a GE interface is encapsulated with Point-to-Point Protocol (PPP) and obtains the remote IP address through PPP negotiation, you need to specify only the outbound interface rather than the next hop address.

- Non-Broadcast Multiple-Access (NBMA) interfaces are applicable to Point-to-Multipoint networks. Therefore, you need to configure IP routes and the mappings between IP addresses and link layer addresses. In this case, next hop IP addresses need to be configured.

- An Ethernet interface is a broadcast interface and a virtual-template (VT) interface can be associated with multiple virtual access interfaces. If the Ethernet interface or the VT interface is specified as the outbound interface of a static route, the next hop cannot be determined because multiple next hops exist. Therefore, do not specify an Ethernet interface or a VT interface as the outbound interface unless necessary. If you need to specify a broadcast interface (such as an Ethernet interface), a VT interface, or an NBMA interface as the outbound interface, you are recommended to specify the associated next hop address at the same time.

- Other attributes

You can configure different preferences for different static routes so that routing management policies can be flexibly applied. For example, when creating multiple routes to the same destination address, you can set the same preference for these routes to implement load balancing. You can also set different preferences to implement routing redundancy.

By configuring different tag values, you can classify static routes to implement different routing policies. For example, other protocols can import static routes with specified tag values based on routing policies.

If service traffic needs to be forwarded along a specified path, regardless of the link status, you can configure permanent advertisement of static routes by using **permanent**.

In network maintenance scenarios, static routes are required to verify services. If you do not want these static routes to be imported by other protocols, specify **no-advertise** to prevent these static routes from being advertised. If you set the destination address and the mask to all 0s (0.0.0.0 0.0.0.0) in the **ip route-static** command, a default route is configured.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Configure an IPv4 static route.

- Run one of the following commands to configure an IPv4 static route for the public network:
 - **ip route-static ip-address { mask | mask-length } { nexthop-address | nexthop6-address } [preference preference | tag tag] * [description text]**
 - **ip route-static ip-address { mask | mask-length } { interface-name | interface-type interface-number } [{ nexthop-address | nexthop6-address }] [preference preference | tag tag] * [description text]**
 - **ip route-static ip-address { mask | mask-length } vpn-instance vpn-instance-name [preference preference | tag tag] * description text**
 - **ip route-static ip-address { mask | mask-length } vpn-instance vpn-instance-name { nexthop-address | nexthop6-address } [preference preference | tag tag] * [description text]**
- Run one of the following commands to configure an IPv4 static route for a VPN instance:
 - **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { interface-type interface-number } [nexthop-address] [preference preference | tag tag] * [description text]**
 - **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } nexthop-address [preference preference | tag tag] * [description text]**
 - **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { public | vpn-instance vpn-destination-name } [preference preference | tag tag] * [description text]**
 - **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } vpn-instance vpn-destination-name nexthop-address [preference preference | tag tag] * [description text]**
- To configure an IPv4 static route in the topology instance, run the **ip route-static topology topology-name ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] } [preference preference | tag tag] * [no-advertise | no-install] [description text]** command.

NOTE

If the outbound interface of a static route is a broadcast interface or an NBMA interface, the next hop of the outbound interface must be specified.

Step 3 Run commit

The configuration is committed.

----End

(Optional) Setting the Default Priority for IPv4 Static Routes

You can change the default priority for IPv4 static routes.

Context

After an IPv4 static route is configured, the default priority is used if no priority is specified for the static route. After the default priority is re-set, the new default priority takes effect only on new IPv4 static routes.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ip route-static default-preference *preference*

The default preference value is configured for static routes.

By default, the priority of IPv4 static routes is lower than those of OSPF and IS-IS routes. To allow IPv4 static routes to take effect when OSPF or IS-IS routes exist, configure the default priority of IPv4 static routes to be higher than that of OSPF or IS-IS routes before you configure IPv4 static routes. A smaller preference value indicates a higher priority.

Step 3 Run commit

The configuration is committed.

----End

(Optional) Configuring Recursion Depth-based Static Route Selection

To prevent inter-board service transmission or routing loop in a static route scenario, configure recursion depth-based static route selection.

Context

Among static routes with the same prefix but different recursion depths, the static routes with shorter recursion depths are more stable. After recursion depth-based static route selection is configured, the system selects the static routes with shorter recursion depths as active routes and delivers them to the Forwarding Information Base (FIB) table. The other routes become inactive.

To prevent inter-board service transmission or routing loop in a static route scenario, configure recursion depth-based static route selection.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip route-static selection-rule relay-depth**

Recursion depth-based static route selection is configured.

Step 3 Run **commit**

The configuration is committed.

----End

(Optional) Configuring a Device to Iterate Static Routes to ARP Vlink Routes

To prevent traffic loss caused by a black-hole route in a scenario where a Layer 2 VPN accesses a Layer 3 VPN, configure devices to iterate static routes to ARP Vlink routes.

Context

Configuring a device to iterate static routes to ARP Vlink routes prevents traffic loss caused by a black-hole route in a scenario where a Layer 2 VPN accesses a Layer 3 VPN.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip route recursive-lookup arp vlink-direct-route protocol static**

The device is configured to iterate static routes to ARP Vlink routes.

In a scenario where a Layer 2 VPN accesses a Layer 3 VPN, to configure a device to recurse static routes to ARP Vlink routes, run the **ip route-static** command with **recursive-lookup host-route** specified and **ip route recursive-lookup arp vlink-direct-route protocol static**.

Step 3 Run **commit**

The configuration is committed.

----End

(Optional) Enabling a Device to Recurse Static Routes to SRv6 Routes

By default, a device cannot recurse static routes to SRv6 routes. In an L3VPNv4 HoVPN over SRv6 BE, L3VPNv4 HoVPN over SRv6 TE Policy, EVPN L3VPN HoVPN over SRv6 BE, or EVPN L3VPN HoVPN over SRv6 TE Policy scenario, you can configure static routes to recurse to SRv6 routes to prevent traffic black holes.

Context

In an L3VPNv4 HoVPN over SRv6 BE, L3VPNv4 HoVPN over SRv6 TE Policy, EVPN L3VPN HoVPN over SRv6 BE, or EVPN L3VPN HoVPN over SRv6 TE Policy scenario,

if the next hop of a default static route on the SPE is the SPE itself and the link between the SPE and the NPE fails, the UPE cannot detect the link fault. As a result, a traffic black hole occurs after traffic reaches the SPE. To resolve this problem, you need to specify the NPE's address (public or private) as the next-hop address of the default static route on the SPE. In this manner, the validity of the static route depends on whether the link between the SPE and NPE is available. In addition, you need to enable the SPE to recurse static routes to SRv6 routes. This configuration allows the SPE to withdraw this inactive route when the SPE-NPE link fails so that the UPE can detect the route withdrawal and no longer sends traffic to the SPE, thereby preventing a traffic black hole.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip route-static recursive-lookup inherit-label-route segment-routing-ipv6**

The device is enabled to recurse IPv4 static routes to SRv6 routes.

Step 3 Run **commit**

The configuration is committed.

----End

(Optional) Preventing an IPv4 Static Route from Being Selected If the BFD Session Associated with It Is in the AdminDown State

This section describes how to configure the router not to select an IPv4 static route if the BFD session associated with it is in the AdminDown state. This ensures that Huawei devices can interwork with non-Huawei devices.

Context

By default, an IPv4 static route can still be selected by Huawei devices even though the BFD session associated with it is in the AdminDown state, but not by non-Huawei devices. As a result, Huawei devices cannot interwork with non-Huawei devices.

To address this problem, run the **ip route-static track bfd-session admindown invalid** command to configure the router not to select the IPv4 static route if the BFD session associated with it is in the AdminDown state.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip route-static track bfd-session session-name *bfd-name* admindown invalid**

The router has been configured not to select the IPv4 static route if the BFD session associated with it is in the AdminDown state.

Step 3 Run commit

The configuration is committed.

----End

Verifying the IPv4 Static Route Configuration

After configuring an IPv4 static route, verify the configuration.

Prerequisites

An IPv4 static route has been configured.

Procedure

- Run the **display ip routing-table** command to check brief information about the IPv4 routing table.
- Run the **display ip routing-table verbose** command to check detailed information about the IPv4 routing table.
- Run the **display ip routing-table protocol** command to check information about routes of a specified routing protocol.

----End

1.1.2.2.4 Configuring IPv4 Floating Static Routes

Configuring IPv4 floating static routes helps to implement route backup and improve network reliability.

Usage Scenario

If there is a route to a specific destination, you can configure a static route with a low preference to implement route backup, which improves network reliability. This static route with a lower preference is called a floating static route. It is activated to forward packets only when the optimal route fails. When the optimal route recovers, this static route becomes inactive.

Floating static routes are used as follows:

- Two static routes with the same destination address but different priorities are configured, and the one with the lower priority functions as a backup.
- When a route of a dynamic routing protocol to a destination address exists, you can configure a static route to the destination address for data service forwarding temporarily if the dynamic routing protocol restarts.

Pre-configuration Tasks

Before configuring IPv4 floating static routes, configure link layer protocol parameters and IP addresses for interfaces and ensure that the status of the link layer protocol on the interfaces is Up.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run either of the following commands as required:

- To configure an IPv4 floating static route on the public network, run the **ip route-static ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag]* [description text]** command.
- To configure an IPv4 floating static route on a VPN, run the **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag]* [description text]** command.
- To configure an IPv4 floating static route in a topology instance, run the **ip route-static topology topology-name ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] } [preference preference | tag tag]* [description text]** command.

The **preference** parameter specifies a route priority. The greater the value, the lower the priority. Therefore, make sure that the priority value of the floating static route is greater than that of the primary route.

Step 3 Run commit

The configuration is committed.

----End

Checking the Configurations

After the configuration is complete, check the configuration results.

- Run the **display ip routing-table ip-address [mask | mask-length]** command to check configurations about the optimal IPv4 static route to the specified destination network segment.
- Run the **display current-configuration | include static** command to check the configurations about the current static route.

1.1.2.2.5 Configuring Association between LDP and Static Routes

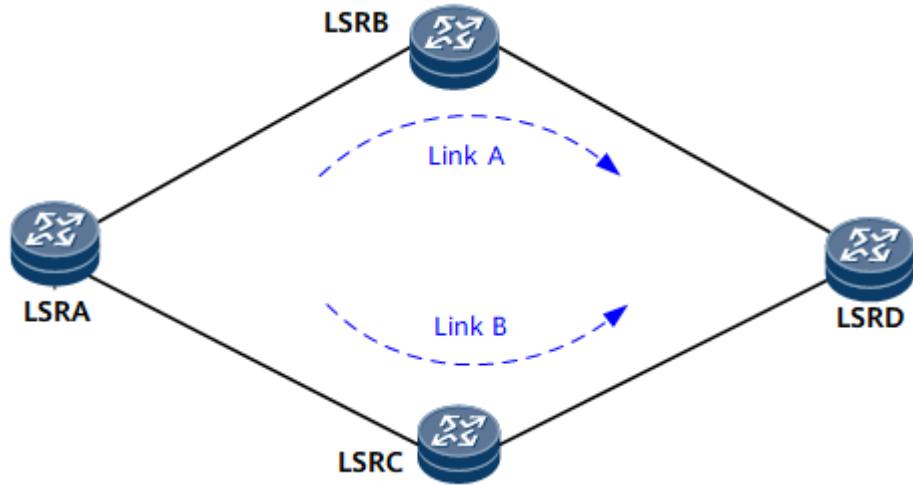
On an MPLS network with primary and secondary LSPs established by label switched routers (LSRs) using static routes, association between LDP and static routes prevents MPLS traffic from being interrupted during traffic switching between the primary and secondary LSPs.

Usage Scenario

On an MPLS network with primary and secondary LSPs established by LSRs using static routes, association between LDP and static routes prevents traffic from being interrupted during traffic switchover from the primary LSP to the secondary

LSP when the LDP session on the primary link fails (not because of a link fault) or during traffic switchback when the primary link recovers from a fault. In [Figure 1-41](#), LSRA and LSRD interwork through static routes. Primary and secondary static routes are deployed on LSRA, with the next-hop devices being LSRB and LSRC, respectively. Primary and secondary LDP LSPs are established based on the static routes. The primary LSP uses Link A, and the secondary LSP uses Link B. In normal cases, Link A is preferred. Association between LDP and static routes in switchover and switchback scenarios is described as follows.

Figure 1-41 Networking for association between LDP and static routes



Switchover scenario

In the switchover scenario, traffic of the primary static route is not switched to the secondary link when the LDP session on the primary link fails (not because of a link fault). As a result, traffic on the LSP over the primary link is interrupted.

After an LDP session is established, LSP traffic travels along the primary link, Link A (LSRA → LSRB → LSRD). If the LDP session between LSRA and LSRB is interrupted, traffic of the primary LSP is switched immediately to the secondary link, Link B (LSRA → LSRC → LSRD). However, because the link between LSRA and LSRB is normal, traffic of the primary static route is not switched to the secondary link. The asynchronous state between LDP and the primary static route causes an LSP traffic interruption.

If association between LDP and static routes is enabled, traffic is automatically switched to the secondary link when the LDP session goes Down, ensuring uninterrupted traffic forwarding.

Switchback scenario

In the switchback scenario, when the primary link recovers from a fault, the traffic of the primary static route is switched back to Link A earlier than the traffic of the primary LSP because the convergence of static routes is faster than that of LDP LSPs. As a result, the secondary LSP on Link B cannot be used, and the LSP on Link A has not been set up yet. As a result, LSP traffic is interrupted.

If the link between LSRA and LSRB fails, traffic is switched immediately to the secondary link, Link B (LSRA → LSRC → LSRD). After the link between LSRA and LSRB recovers, traffic of the primary static route is immediately switched back to

Link A (LSRA → LSRB → LSRD). However, the secondary LSP cannot be used, and the LSP on Link A has not recovered yet. As a result, traffic is interrupted.

If association between LDP and static routes is enabled, the static route on Link A becomes active only when the LDP session on Link A goes Up. In this manner, the states of the primary static route and LSP are asynchronous during the switchback, which prevents traffic loss.

 NOTE

Only the static routes with outbound interfaces specified can be associated with LDP.

Pre-configuration Tasks

Before configuring association between LDP and static routes, complete the following tasks:

- Enable MPLS.
- Enable MPLS LDP globally and on interfaces.
- Ensure that LDP sessions are set up between devices.

Procedure

- Enable association between LDP and static routes.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **ip route-static [vpn-instance vpn-instance-name] ip-address { mask | mask-length } interface-type interface-number [next-hop-address] [preference preference | tag tag] * ldp-sync [description text]**
Association between LDP and static routes is enabled.
 - c. Run **commit**
The configuration is committed.
- (Optional) Set a Hold-down timer.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface interface-type interface-number**
The outbound interface view of the primary link of the static route is displayed.
 - c. Run **static-route timer ldp-sync hold-down { timer | infinite }**
A period in which the static route is inactive and waits for the establishment of an LDP session is set.
 - d. Run **commit**
The configuration is committed.

 NOTE

The preceding configurations need to be performed on both ends of the primary and secondary links.

----End

Checking the Configurations

After configuring association between LDP and static routes, run the **display static-route ldp-sync** command.

1.1.2.2.6 Configuring IPv4 Static Route Detection

With IPv4 static route detection, if a link through which an IPv4 static route travels fails, a rapid link switchover is performed, preventing lengthy service interruptions.

Usage Scenario

Unlike dynamic routes, static routes do not have a detection mechanism. As a result, administrator intervention is required if a fault occurs on the network. Static route detection methods can be used to quickly detect link faults. If a link fails, a link switchover is performed quickly to prevent lengthy service interruption.

The following IPv4 static route detection methods are available:

- **BFD for IPv4 static routes:** After BFD for IPv4 static routes is configured, each static route can be bound to a BFD session. BFD for IPv4 static routes implements millisecond-level detection. BFD is classified as static BFD or dynamic BFD.
- **Network quality analysis (NQA) for IPv4 static routes:** Although BFD for IPv4 static routes implements fast link fault detection, it cannot be deployed in some scenarios (when Layer 2 devices exist on the network, for example) because both ends of the link must support BFD. NQA for IPv4 static routes, however, can monitor the link status of a static route even when only one end supports NQA. NQA for IPv4 static routes implements second-level detection.
- **Association between Ethernet in the First Mile (EFM) and IPv4 static routes:** After EFM OAM is enabled, EFM can be associated with IPv4 static routes. EFM for IPv4 static routes enables the system to respond to interface up or down events and determine whether to activate static routes. This mechanism controls route advertisement and ensures that the traffic from the remote end can be correctly forwarded.



Only one of the preceding static route detection methods can be deployed. Therefore, choose one based on the live network requirements.

Pre-configuration Tasks

Before configuring IPv4 static route detection, configure link-layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol of each interface is up.

Configuring Dynamic BFD for IPv4 Static Routes

Dynamic BFD for IPv4 static routes enables devices to fast detect link changes, improving network reliability.

Usage Scenario

To use BFD sessions to provide link detection for IPv4 static routes on the public network, you can bind IPv4 static routes to BFD sessions. One IPv4 static route can be bound to one BFD session.

Optimal IPv4 static routes are delivered to the forwarding table for packet forwarding. However, IPv4 static routes cannot detect the status of the link to the next hop. You can bind IPv4 static routes to BFD sessions. A BFD session can fast detect changes over a link and inform the routing management system of the changes. The routing management system immediately deletes the IPv4 static route that is bound to the BFD session from the forwarding table and recalculates another active route. In this manner, fast route convergence is implemented.

Pre-configuration Tasks

Before configuring dynamic BFD for static routes, configure parameters of the link layer protocol and IP addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bfd**

Global BFD is enabled on the local device.



Running the **undo bfd** command will delete the parameters of the BFD session bound to the static route. As a result, the static route status may change, and services may be interrupted.

Step 3 Run **quit**

Return to the system view.

Step 4 (Optional) Run **ip route-static default-bfd [min-rx-interval min-rx-interval] [min-tx-interval min-tx-interval] [detect-multiplier multiplier] ***

Global BFD parameters are configured for IPv4 static routes.

Step 5 Run **ip route-static bfd [interface-type interface-number | vpn-instance vpn-instance-name] nexthop-address [local-address address] [min-rx-interval min-rx-interval] [min-tx-interval min-tx-interval] [detect-multiplier multiplier] ***

The BFD parameters are set for an IPv4 static route.

 NOTE

If *interface-type interface-number* is not set, **local-address** *address* must be specified.

If *interface-type interface-number* is specified, the BFD session monitors the active status of the corresponding link on the specified interface. If **local-address** *address* is specified, the BFD session monitors the active status of the corresponding link.

If none of **min-rx-interval**, **min-tx-interval**, or **detect-multiplier** is specified, the default values of the global BFD parameters are used.

Step 6 Run either of the following commands as required:

- Run **ip route-static** *ip-address* { *mask* | *mask-length* } { *nexthop-address* | *interface-type interface-number* [*nexthop-address*] | **vpn-instance** *vpn-instance-name* *nexthop-address* } **bfd enable** [**description** *text*]
A public network IPv4 static route is bound to a BFD session.
- Run **ip route-static** **vpn-instance** *vpn-source-name* *destination-address* { *mask* | *mask-length* } { *nexthop-address* | *interface-type interface-number* [*nexthop-address*] | **vpn-instance** *vpn-instance-name* *nexthop-address* } **bfd enable** [**description** *text*]
A VPN network IPv4 static route is bound to a BFD session.

 NOTE

The outbound interface and next hop IP address specified when binding the IPv4 static route to a BFD session must be the same as those specified when configuring BFD parameters for the IPv4 static route.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the following commands to check the previous configuration.

- Run the **display bfd session** { **all** | **discriminator** *discr-value* } [**verbose**] command to check detailed BFD session information.
- Run the **display current-configuration** | **include bfd** command to check configurations of BFD for static routes.

You can check the information about a BFD session only after parameters of the BFD session are set and the BFD session is established.

If a BFD session has been established, the status of the BFD session is Up. Run the **display current-configuration** | **include bfd** command in the system view, and you can view that the BFD session has been bound to a static route.

Configuring Static BFD for IPv4 Static Routes

Static BFD for IPv4 static routes enables devices to fast detect link changes, improving network reliability.

Usage Scenario

To use BFD sessions to provide link detection for IPv4 static routes on the public network, you can bind IPv4 static routes to BFD sessions. One IPv4 static route can be bound to one BFD session.

Optimal IPv4 static routes are delivered to the forwarding table for packet forwarding. However, IPv4 static routes cannot detect the status of the link to the next hop. You can bind IPv4 static routes to BFD sessions. A BFD session can fast detect changes over a link and inform the routing management system of the changes. The routing management system immediately deletes the IPv4 static route that is bound to the BFD session from the forwarding table and recalculates another active route. In this manner, fast route convergence is implemented.

Pre-configuration Tasks

Before configuring static BFD for IPv4 static routes, complete the following tasks:

- Configure parameters of the link layer protocol and IP addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.

Procedure

- Configure a BFD Session.
 - Run **system-view**
The system view is displayed.
 - Run **bfd**
BFD is globally enabled.
 - Run **quit**
The system view is displayed.
 - Run **bfd session-name bind peer-ip peer-ip**
The binding between a BFD session and a peer IP address is created and the BFD session view is displayed.
 - Run **discriminator local discr-value**
The local discriminator of a static BFD session is set.
 - Run **discriminator remote discr-value**
The remote discriminator of a static BFD session is set.

NOTE

For details of optional procedures when configuring BFD session, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - Network Reliability*.

- Run **commit**
The configuration is committed.
- Associating an IPv4 Static Route with a BFD Session.
 - Run **system-view**
The system view is displayed.

b. Run either of the following commands as required:

- Run **ip route-static ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } track bfd-session cfg-name [description text]**
A BFD session is bound to an IPv4 static route on the public network.
- Run **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } track bfd-session cfg-name [description text]**
A BFD session is bound to an IPv4 static route on the VPN network.

c. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the following commands to check the previous configuration.

- Run the **display bfd session { all | discriminator descr-value } [verbose]** command to check information about the BFD session.
- Run the **display current-configuration | include bfd** command to check the configuration of BFD for static routes.

You can check the information about a BFD session only after parameters of the BFD session are set and the BFD session is established.

If a BFD session has been established, the status of the BFD session is Up. Run the **display current-configuration | include bfd** command in the system view, and you can see that the BFD session has been bound to a static route.

Configuring NQA for IPv4 Static Routes

If an IPv4 static route is associated with a network quality analysis (NQA) test instance, NQA tests the link status periodically. If NQA detects a fault along the associated IPv4 static route, the IPv4 static route is deleted, and traffic is switched to another route.

Usage Scenario

On live networks, the link status of IPv4 static routes must be monitored in real time so that a link switchover can be performed immediately if a link fails. Bidirectional Forwarding Detection (BFD) for IPv4 static routes can implement detection within milliseconds. However, BFD for IPv4 static routes requires that both ends of a link support BFD.

NQA for IPv4 static routes can monitor the link status of static routes as long as one end supports NQA.

If a link fails, an NQA test instance immediately detects the fault and instructs the routing management module to delete the IPv4 static route associated with the

NQA test instance from the IP routing table. Traffic is then forwarded over another link.

 NOTE

Currently, NQA test instances of only the ICMP or TCP type can be associated with static routes to implement fast fault detection.

Pre-configuration Tasks

Before configuring NQA for IPv4 static routes, configure parameters of the link layer protocol and IP addresses for interfaces and ensure that the link layer protocol on the interfaces is up.

Procedure

- Configure an NQA test instance of the ICMP or TCP type.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **nqa test-instance admin-name test-name**
An NQA test instance is created, and the test instance view is displayed.

 NOTE

Running the **undo nqa all-test-instance** command will delete the parameters of the NQA test instance bound to the static route. As a result, the static route status may change, and services may be interrupted.

- c. Run **test-type { icmp | tcp }**
The type of the test instance is set to ICMP or TCP.
- d. Run **destination-address ipv4 destAddress**
A destination address is configured.

In an NQA test instance, you can specify an NQA server by running the **destination-address** command to configure a destination address for the NQA test instance.

- e. Run any of the following commands to start an NQA test instance:

- **start at [yyyy/mm/dd] hh:mm:ss [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start delay { seconds second | hh:mm:ss } [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start now [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start daily hh:mm:ss to hh:mm:ss [begin { yyyy/mm/dd | yyyy-mm-dd }] [end { yyyy/mm/dd | yyyy-mm-dd }]**

 NOTE

For details of optional procedures when configuring an ICMP NQA test instance, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - System Monitor*.

f. Run **commit**

The configuration is committed.

- Associate an IPv4 static route with an NQA test instance.

a. Run **system-view**

The system view is displayed.

- Perform either of the following operations to associate an IPv4 static route with the NQA test instance.

- To configure an IPv4 static route on the public network and associate the route with an NQA test instance, run the **ip route-static ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address | vpn-instance vpn-destination-name }** [**preference preference | tag tag**] * **track nqa admin-name test-name [description text]** command.
- To configure an IPv4 static route on a VPN and associate the route with an NQA test instance, run the **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { nexthop-address [public] | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address | vpn-instance vpn-destination-name | public }** [**preference preference | tag tag**] * **track nqa admin-name test-name [description text]** command.

c. Run **commit**

The configuration is committed.

 NOTE

Precautions for associating a static route with an NQA test instance are as follows:

- The destination address of an NQA test instance cannot be the destination address of an associated static route.
- If the static route associated with one NQA test instance is associated with another NQA test instance, the association between the static route and the original NQA test instance is deleted.
- An NQA test instance must have been created before you associate it with a static route.

----End

Verifying the Configuration

Run the following commands to check the previous configuration.

- Run the **display current-configuration | include nqa** command to check the configurations of NQA for static routes.

- Run the **display nqa results [test-instance admin-name test-name]** command to check the NQA test results.

 NOTE

The NQA test results cannot be displayed automatically. You need to run the **display nqa results** command to view the results.

Configuring NQA Group for IPv4 Static Route

After an IPv4 static route is associated with an NQA group, if the NQA group detects a link fault, it withdraws the IPv4 static route, triggering traffic switching and ensuring network stability.

Usage Scenario

On live networks, the status of IPv4 static routes needs to be monitored in real time to ensure network stability. If the link status changes, a primary/backup link switchover is performed. NQA for IPv4 static route can monitor only a single link. To enable a static route to use NQA test instances to monitor multiple links, you can bind these NQA test instances to the same NQA group and associate the static route with the NQA group.

If a link fault occurs, the NQA group can collect statistics on the status changes of all NQA test instances bound to the NQA group and determine whether to change its status according to the configured operation. When the status of the NQA group changes, the NQA group instructs the RM module to delete the IPv4 static route associated with the NQA group from the IP routing table. Traffic is then forwarded along another path, implementing a link switchover.

Pre-configuration Tasks

Before configuring NQA group for IPv4 static route, complete the following task:

- Configure link-layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol of each interface is up.

Procedure

- Configure multiple NQA test instances of the ICMP or TCP type.
 - Run **system-view**
The system view is displayed.
 - Run **nqa test-instance admin-name test-name**
An NQA test instance is created, and the test instance view is displayed.
 - Run **test-type { icmp | tcp }**
The test instance type is set to ICMP or TCP.
 - Run **destination-address ipv4 destAddress**
A destination address is configured.
In an NQA test instance, you can specify the destination end by running the **destination-address** command to configure a destination address for the NQA test instance.
 - Run any of the following commands to start an NQA test instance:

- **start at [yyyy/mm/dd] hh:mm:ss [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start delay { seconds second | hh:mm:ss } [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start now [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start daily hh:mm:ss to hh:mm:ss [begin { yyyy/mm/dd | yyyy-mm-dd }] [end { yyyy/mm/dd | yyyy-mm-dd }]**

 **NOTE**

For more optional configurations of NQA test instances, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - System Monitor*.

- f. Run **commit**

The configuration is committed.
 - g. Repeat the preceding steps to configure multiple NQA test instances of the ICMP or TCP type.
- Bind the preceding NQA test instances to an NQA group.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **nqa group group-name**

An NQA group is created, and the NQA group view is displayed.
 - c. Run **nqa test-instance admin-name test-name**

An NQA test instance is bound to the NQA group.

Run this command multiple times to bind multiple NQA test instances to the NQA group.

 **NOTE**

The NQA test instances bound to the same NQA group must be of the same type: ICMP or TCP.

- d. Run **operator { and | or }**

The operation type between test instances in the NQA group is set to AND or OR.

By default, the operation type between test instances is **or**.
 - e. (Optional) Run **description string**

A description is configured for the NQA group.
 - f. Run **commit**

The configuration is committed.
- Bind an IPv4 static route to the NQA group.

a. Run **system-view**

The system view is displayed.

b. Bind an IPv4 static route to the NQA group.

- To configure an IPv4 static route on the public network and associate the route with the NQA group, run the **ip route-static ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag] * track nqa-group group-name [description text]** command.
- To configure an IPv4 static route on a VPN and associate the route with the NQA group, run the **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { nexthop-address [public] | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag] * track nqa-group group-name [description text]** command.

c. Run **commit**

The configuration is committed.



When configuring NQA group for IPv4 static route, pay attention to the following points:

- The destination address of an NQA group cannot be the destination address of an associated static route.
- If a static route is associated with another NQA group, the association between the static route and the current NQA group is removed.
- A static route cannot be associated with an NQA group that has not been created.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display static-route routing-table** command to check information about static routes.
- Run the **display current-configuration | include nqa-group** command to check the configurations of NQA group for IPv4 static route.
- Run the **display nqa group [group-name]** command to check the test result of the NQA group.



The test result of the NQA group cannot be displayed automatically. You need to run the **display nqa group** command to view the result.

Associating EFM with IPv4 Static Routes

After EFM is associated with IPv4 static routes, the system determines whether to activate the static routes based on the EFM session status.

Usage Scenario

After EFM is associated with IPv4 static routes, the system responds to EFM Up/Down events on a specified interface and determines whether to activate static routes. In this manner, route advertisement is controlled and remote traffic is forwarded.

On MANs, EFM is mainly used between customer edges (CEs) and underlayer provider edges (UPEs) to ensure the reliability and stability of connections between user networks and carrier networks. EFM monitors and rectifies faults on P2P Ethernet physical links or simulated links. It is recommended that EFM be used on user access networks.

Pre-configuration Tasks

Before associating EFM with IPv4 static routes, complete the following tasks:

- Configure link-layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol of each interface is up.
- Configure EFM OAM. For configuration details, see "Network Reliability" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide*.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Associate EFM with IPv4 static routes.

- To associate EFM with an IPv4 static route on the public network, run the **ip route-static ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag] * track efm-state interface-type interface-number [description text]** command.
- To associate EFM with an IPv4 static route in a VPN instance, run the **ip route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag] * track efm-state interface-type interface-number [description text]** command.

Step 3 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the following commands to check the previous configuration.

- Run the **display efm session { all | interface interface-type interface-number }** command to check information about EFM OAM on a specified interface.

- Run the **display current-configuration | include efm** command to check configurations of association between EFM and IPv4 static routes.

1.1.2.2.7 Configuring FRR for IPv4 Static Routes

Fast reroute (FRR) is applicable to services that are sensitive to packet delay and packet loss. FRR can be configured for IPv4 static routes to protect traffic using backup links.

Usage Scenario

FRR is applicable to IP services that are sensitive to delay and packet loss. FRR minimizes the impact of link faults on services.

You can specify different priorities for different static routes to implement FRR for static routes. Routes with the same destination IP address but different priorities can back up each other.

Pre-configuration Tasks

Before configuring FRR for IPv4 static routes, complete the following tasks:

- Configure parameters of the link layer protocol and IP addresses for interfaces and ensure that the link layer protocol on the interfaces is Up.
- Configure multiple routes with the same destination IP address but different priorities.
- Configure dynamic BFD for IPv4 static routes** or **configure static BFD for IPv4 static routes** to speed up fault detection.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip route-static frr [vpn-instance vpn-instance-name]**

FRR is enabled for public or VPN IPv4 static routes.

NOTE

To implement route backup by configuring FRR for static routes, specify different priorities for these static routes.

If FRR for static routes and BFD are both configured and a static route has an Ethernet interface as its outbound interface but has no next hop address, FRR cannot be implemented between this static route and those with next hop addresses. To implement FRR between them, specify a next hop address for this route.

Step 3 Run **commit**

The configuration is committed.

----End

Checking the Configurations

Run the following commands to check the previous configuration.

- Run the **display ip routing-table verbose** command to check detailed information about the backup outbound interface and the backup next hop in the routing table.
- Run the **display ip routing-table ip-address [mask | mask-length] [longer-match] verbose** command to check detailed information about the backup outbound interface and the backup next hop in the routing table.
- Run the **display ip routing-table ip-address1 { mask1 | mask-length1 } ip-address2 { mask2 | mask-length2 } verbose** command to check detailed information about the backup outbound interface and the backup next hop in the routing table.

1.1.2.2.8 Configuration Examples for IPv4 Static Routes

This section provides configuration examples of IPv4 static routes.

Example for Configuring IPv4 Static Routes

You can configure IPv4 static routes to interconnect any two devices on an IPv4 network.

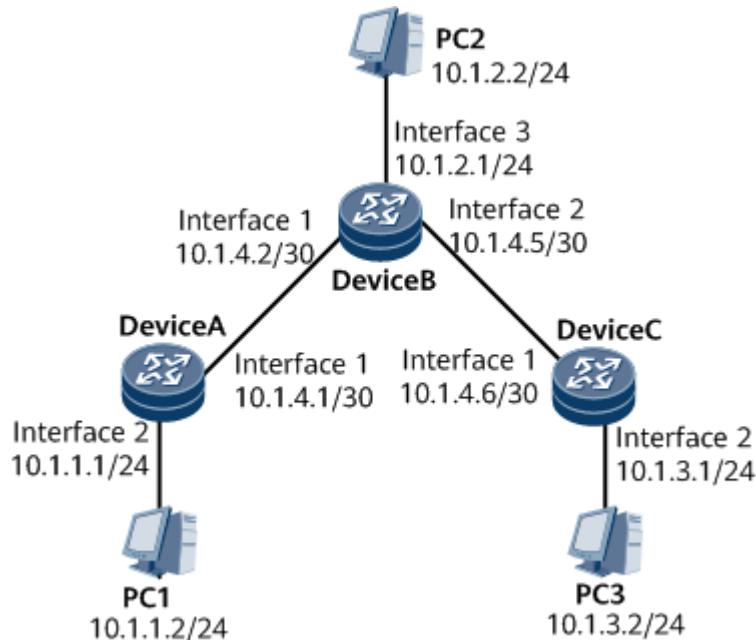
Networking Requirements

Figure 1-42 shows the IP addresses and masks of interfaces and hosts. It is required that any two hosts in **Figure 1-42** communicate through static routes.

Figure 1-42 Networking for configuring IPv4 static routes



Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Precautions

When configuring an IPv4 static route, specify a next-hop address if the outbound interface is of the broadcast type.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IPv4 addresses for interfaces on each router.
2. Configure an IPv4 static route and a default route on each router.
3. Configure the IPv4 default gateway on each host so that any two hosts communicate with each other.

Data Preparation

To complete the configuration, you need the following data:

- Default route with 10.1.4.2 as the next hop on Device A
- Static route to 10.1.1.0 with 10.1.4.1 as the next hop on Device B
- Static route to 10.1.3.0 with 10.1.4.6 as the next hop on Device B
- Default route with 10.1.4.5 as the next hop on Device C
- Default gateways of PC1, PC2, and PC3

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure static routes.

Configure an IPv4 default route on Device A.

```
[~DeviceA] ip route-static 0.0.0.0 0.0.0.0 10.1.4.2  
[*DeviceA] commit
```

Configure two IPv4 static routes on Device B.

```
[~DeviceB] ip route-static 10.1.1.0 255.255.255.0 10.1.4.1  
[*DeviceB] ip route-static 10.1.3.0 255.255.255.0 10.1.4.6  
[*DeviceB] commit
```

Configure an IPv4 default route on Device C.

```
[~DeviceC] ip route-static 0.0.0.0 0.0.0.0 10.1.4.5  
[*DeviceC] commit
```

Step 3 Configure a default gateway for each host.

Configure the default gateways of PC1, PC2, and PC3 as 10.1.1.1, 10.1.2.1, and 10.1.3.1 respectively.

Step 4 Verify the configuration.

Check the IP routing table of Device A.

```
[~DeviceA] display ip routing-table  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

Routing Table: Public						
Destinations : 12			Routes : 12			
Destination/Mask	Proto	Pre	Cost	Flags	NextHop	Interface
0.0.0.0/0	Static	60	0	RD	10.1.4.2	GigabitEthernet1/0/0
10.1.1.0/24	Direct	0	0	D	10.1.1.1	GigabitEthernet2/0/0
10.1.1.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet2/0/0
10.1.1.255/32	Direct	0	0	D	127.0.0.1	GigabitEthernet2/0/0
10.1.4.0/30	Direct	0	0	D	10.1.4.1	GigabitEthernet1/0/0
10.1.4.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet1/0/0
10.1.4.2/32	Direct	0	0	D	10.1.4.2	GigabitEthernet1/0/0
10.1.4.255/32	Direct	0	0	D	127.0.0.1	GigabitEthernet1/0/0
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0
127.0.0.1/32	Direct	0	0	D	127.0.0.1	InLoopBack0
127.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0
255.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0

Run the **ping** command to verify the connectivity.

```
[~DeviceA] ping 10.1.3.1
PING 10.1.3.1: 56 data bytes, press CTRL_C to break
Reply from 10.1.3.1: bytes=56 Sequence=1 ttl=254 time=62 ms
Reply from 10.1.3.1: bytes=56 Sequence=2 ttl=254 time=63 ms
Reply from 10.1.3.1: bytes=56 Sequence=3 ttl=254 time=63 ms
Reply from 10.1.3.1: bytes=56 Sequence=4 ttl=254 time=62 ms
Reply from 10.1.3.1: bytes=56 Sequence=5 ttl=254 time=62 ms
--- 10.1.3.1 ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 62/62/63 ms
```

Run the **tracert** command to verify the connectivity.

```
[~DeviceA] tracert 10.1.3.1
traceroute to 10.1.3.1(10.1.3.1), max hops: 30 ,packet length: 40
1 10.1.4.2 31 ms 32 ms 31 ms
2 10.1.4.6 62 ms 63 ms 62 ms
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.1 255.255.255.252
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
#
ip route-static 0.0.0.0 0.0.0.0 10.1.4.2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.2 255.255.255.252
#
```

```
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.4.5 255.255.255.252
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
#
ip route-static 10.1.1.0 255.255.255.0 10.1.4.1
ip route-static 10.1.3.0 255.255.255.0 10.1.4.6
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.6 255.255.255.252
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0
#
ip route-static 0.0.0.0 0.0.0.0 10.1.4.5
#
return
```

Example for Configuring IPv4 Floating Static Routes

IPv4 Floating static routes can be used for the static route backup.

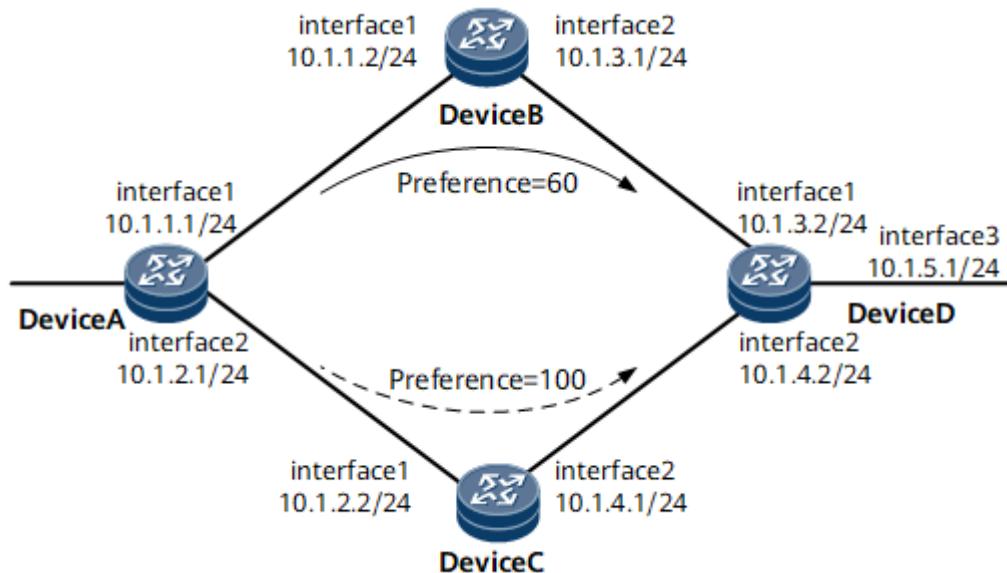
Networking Requirements

Figure 1-43 shows the IP addresses and masks of each router interface. Two IPv4 static routes to 10.1.5.0/24 are configured on Device A. The primary static route passes through Device B, and the floating static route passes through Device C.

Figure 1-43 Networking for configuring IPv4 floating static routes



Interfaces 1 through 3 in this example represent GE 1/0/1, GE 1/0/2, and GE 1/0/3, respectively.



Precautions

When configuring an IPv4 floating static route, a next-hop address of this route must be specified if the outbound interface is of the broadcast type.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an IPv4 address for each interface of each router.
2. On Devices B and C, configure IPv4 static routes to 10.1.5.0/24.
3. On Device A, configure two IPv4 static routes to 10.1.5.0/24 with different priorities.
4. On Device D, configure IPv4 static routes to 10.1.1.0/24 and 10.1.2.0/24 so that routers can communicate.

Data Preparation

To complete the configuration, you need the following data:

- On Device A, priority values of two static routes (60 for the one with 10.1.1.2 as the next hop address and 100 for the one with 10.1.2.2 as the next-hop address)

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration File](#) in this section.

Step 2 Configure IPv4 static routes.

Configure IPv4 static routes on Device B.

```
[~DeviceB] ip route-static 10.1.5.0 24 10.1.3.2
[*DeviceB] commit
```

Configure IPv4 static routes on Device C.

```
[~DeviceC] ip route-static 10.1.5.0 24 10.1.4.2  
[*DeviceC] commit
```

Configure IPv4 static routes on Device A.

```
[*DeviceA] ip route-static 10.1.5.0 24 10.1.1.2  
[*DeviceA] ip route-static 10.1.5.0 24 10.1.2.2 preference 100  
[*DeviceA] commit
```

Configure IPv4 static routes on Device D.

```
[~DeviceD] ip route-static 10.1.1.0 24 10.1.3.1  
[*DeviceD] ip route-static 10.1.2.0 24 10.1.4.1  
[*DeviceD] commit
```

Step 3 Verify the configuration.

View information about static routes in the IP routing table of Device A.

```
<DeviceA> display ip routing-table protocol static  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route  
-----  
_public_ Routing Table : Static  
  
Destinations : 1      Routes : 1      Configured Routes : 1  
Static routing table status : <Active>  
    Destinations : 1      Routes : 1  
  
Destination/Mask   Proto Pre Cost      Flags NextHop      Interface  
10.1.5.0/24  Static 60  0          RD 10.1.1.2      GigabitEthernet1/0/1  
  
Static routing table status : <Inactive>  
    Destinations : 0      Routes : 0
```

Use the **tracert** command to check the connectivity on Device A.

```
<DeviceA> tracert 10.1.5.1  
traceroute to 10.1.5.1(10.1.5.1), max hops: 30 ,packet length: 40  
1 10.1.1.2 90 ms 1 ms 1 ms  
2 10.1.5.1 4 ms 1 ms 2 ms
```

Run the **shutdown** command on GE 1/0/1 of Device A to simulate a link fault.

```
[~DeviceA] interface GigabitEthernet 1/0/1  
[~DeviceA-GigabitEthernet1/0/1] shutdown  
[*DeviceA-GigabitEthernet1/0/1] commit
```

View information about static routes in the IP routing table of Device A. The route to 10.1.5.0/24 switches to the floating static route with next hop 10.1.2.2.

```
[~DeviceA-GigabitEthernet1/0/1] quit  
[~DeviceA] quit  
<DeviceA> display ip routing-table protocol static  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route  
-----  
_public_ Routing Table : Static  
  
Destinations : 1      Routes : 1      Configured Routes : 1  
Static routing table status : <Active>  
    Destinations : 1      Routes : 1  
  
Destination/Mask   Proto Pre Cost      Flags NextHop      Interface  
10.1.5.0/24  Static 100  0          RD 10.1.2.2      GigabitEthernet1/0/2  
  
Static routing table status : <Inactive>  
    Destinations : 0      Routes : 0
```

Use the **tracert** command to check the connectivity on Device A.

```
<DeviceA> tracert 10.1.5.1
traceroute to 10.1.5.1(10.1.5.1), max hops: 30 ,packet length: 40
1 10.1.2.2 100 ms 1 ms 1 ms
2 10.1.5.1 5 ms 1 ms 2 ms
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.2.1 255.255.255.0
#
ip route-static 10.1.5.0 24 10.1.1.2
ip route-static 10.1.5.0 24 10.1.2.2 preference 100
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.3.1 255.255.255.0
#
ip route-static 10.1.5.0 24 10.1.3.2
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.4.1 255.255.255.0
#
ip route-static 10.1.5.0 24 10.1.4.2
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.3.2 255.255.255.0
#
```

```
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 10.1.5.1 255.255.255.0
#
ip route-static 10.1.1.0 24 10.1.3.1
ip route-static 10.1.2.0 24 10.1.4.1
#
return
```

Example for Configuring Dynamic BFD for IPv4 Static Routes

Dynamic BFD for IPv4 static routes can fast detect link failures.

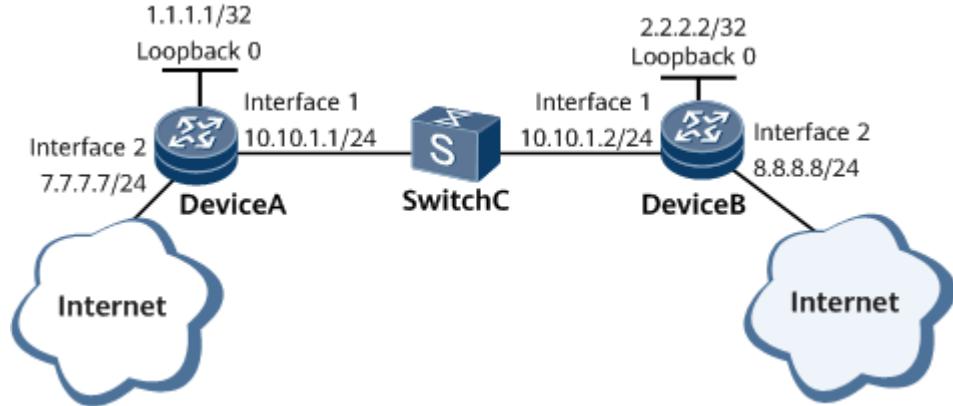
Networking Requirements

On the network shown in [Figure 1-44](#), DeviceA is connected to DeviceB through SwitchC. It is required that DeviceA communicate with other Devices through static default routes and that a BFD session be established between DeviceA and DeviceB to detect link failures.

Figure 1-44 Configuring BFD for static routes

 NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Precautions

When configuring dynamic BFD for static routes, note the following:

- BFD has been enabled globally.
- The parameters configured on the two ends of a BFD session must be consistent.

Configuration Roadmap

The configuration roadmap is as follows:

1. On DeviceA, configure an IPv4 static route to DeviceB.
2. Configure dynamic BFD for static routes.

Data Preparation

To complete the configuration, you need the following data:

- Peer IP address to be detected by BFD
- Default values of the local detection multiplier and of the minimum intervals at which BFD Control packets are sent and received

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure static routes.

On DeviceA, configure a static route to 2.2.2.2/32.

```
[~DeviceA] ip route-static 2.2.2.2 32 10.10.1.2  
[*DeviceA] commit
```

Check the IP routing table of DeviceA. The following command output shows that the static route exists in the routing table.

```
[~DeviceA] display ip routing-table  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route  
-----  
Routing Table : _public_  
Destinations : 10      Routes : 10  
  
Destination/Mask Proto Pre Cost     Flags NextHop       Interface  
1.1.1.1/32 Direct 0   0          D 127.0.0.1    LoopBack0  
2.2.2.2/32 Static 60  0          RD 10.10.1.2  GigabitEthernet1/0/0  
127.0.0.0/8 Direct 0   0          D 127.0.0.1    InLoopBack0  
127.0.0.1/32 Direct 0   0          D 127.0.0.1    InLoopBack0  
127.255.255.255/32 Direct 0  0          D 127.0.0.1    InLoopBack0  
255.255.255.255/32 Direct 0  0          D 127.0.0.1    InLoopBack0  
10.10.1.0/24 Direct 0   0          D 10.10.1.1   GigabitEthernet1/0/0  
10.10.1.1/32 Direct 0   0          D 127.0.0.1   GigabitEthernet1/0/0  
10.10.1.2/32 Direct 0   0          D 10.10.1.2   GigabitEthernet1/0/0  
10.10.1.255/32 Direct 0  0          D 127.0.0.1   GigabitEthernet1/0/0
```

On DeviceB, configure a static route to 1.1.1.1/32.

```
[~DeviceB] ip route-static 1.1.1.1 32 10.10.1.1  
[*DeviceB] commit
```

Step 3 Configure dynamic BFD for static routes.

On DeviceA, bind the static route to the BFD session.

```
[~DeviceA] bfd  
[*DeviceA-bfd] quit  
[*DeviceA] ip route-static bfd 10.10.1.2 local-address 10.10.1.1  
[*DeviceA] ip route-static 2.2.2.2 32 10.10.1.2 bfd enable  
[*DeviceA] commit
```

On DeviceB, bind the static route to the BFD session.

```
[~DeviceB] bfd  
[*DeviceB-bfd] quit  
[*DeviceB] ip route-static bfd 10.10.1.1 local-address 10.10.1.2  
[*DeviceB] ip route-static 1.1.1.1 32 10.10.1.1 bfd enable  
[*DeviceB] commit
```

Step 4 Verify the configuration.

After the preceding configuration, on DeviceA and DeviceB, you can view that the BFD session has been established and is Up and that the static routes have been bound to the BFD session.

Use the command output DeviceA as an example.

```
[~DeviceA] display bfd session all verbose
(w): State in WTR
(*): State is invalid

-----  

(Multi Hop) State : Up          Name : dyn_8193  

-----  

Local Discriminator : 8193      Remote Discriminator : 8193  

Session Detect Mode : Asynchronous Mode Without Echo Function  

BFD Bind Type      : Peer IP Address  

Bind Session Type   : Dynamic  

Bind Peer IP Address : 10.10.1.2  

Bind Interface     : -  

Bind Source IP Address : 10.10.1.1  

  FSM Board Id       : 0          TOS-EXP           : 7  

Min Tx Interval (ms) : 50      Min Rx Interval (ms) : 50  

  Actual Tx Interval (ms): 50    Actual Rx Interval (ms): 50  

Local Detect Multi : 3        Detect Interval (ms) : 150  

  Echo Passive       : Disable   Acl Number        : -  

  Destination Port   : 4784      TTL               : 253  

  Proc Interface Status : Disable Process PST       : Disable  

  WTR Interval (ms)   : 0        Local Demand Mode : Disable  

  Active Multi       : 3  

Last Local Diagnostic : No Diagnostic  

Bind Application : STATICRT  

  Session TX TmrID   : 0          Session Detect TmrID : 0  

  Session Init TmrID : -          Session WTR TmrID  : -  

  Session Echo Tx TmrID : -  

  Session Description : -  

  Track Group Name   : -  

-----  

Total UP/DOWN Session Number : 1/0
```

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA  
#  
bfd  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.10.1.1 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 7.7.7.7 255.255.255.0  
#  
interface LoopBack0  
ip address 1.1.1.1 255.255.255.255  
#  
ip route-static bfd 10.10.1.2 local-address 10.10.1.1  
ip route-static 2.2.2.2 32 10.10.1.2 bfd enable  
#  
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
bfd  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.10.1.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 8.8.8.8 255.255.255.0  
#  
interface LoopBack0  
ip address 2.2.2.2 255.255.255.255  
#  
ip route-static bfd 10.10.1.1 local-address 10.10.1.2  
ip route-static 1.1.1.1 32 10.10.1.1 bfd enable  
#  
return
```

Example for Configuring Static BFD for IPv4 Static Routes

To improve network reliability, you can configure static BFD for IPv4 static routes to fast detect link failures and speed up route convergence.

Networking Requirements

On the network shown in [Figure 1-45](#), DeviceA is connected to DeviceB through Switch C. It is required that DeviceA communicate with other devices through static default routes and that a BFD session be set up between DeviceA and DeviceB to detect link faults.

Figure 1-45 Configuring static BFD for IPv4 static routes



Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Configure a BFD session between DeviceA and DeviceB to detect the link between the two devices.
2. Configure a default static route from DeviceA to the external network and bind the default static route to the BFD session.

Data Preparation

To complete the configuration, you need the following data:

- Peer IP address to be detected by BFD
- Local discriminator and remote discriminator of a BFD session
- Default values of the local detection multiplier and of the minimum intervals at which BFD Control packets are sent and received

Procedure

Step 1 Configure an IP address for each interface.

For configuration details, see "Configuration Files" in this section.

Step 2 Configure a BFD session between DeviceA and DeviceB.

On DeviceA, configure a BFD session between DeviceA and DeviceB.

```
<DeviceA> system-view
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bfd aa bind peer-ip 1.1.1.2
[*DeviceA-bfd-session-aa] discriminator local 10
[*DeviceA-bfd-session-aa] discriminator remote 20
[*DeviceA-bfd-session-aa] commit
[~DeviceA-bfd-session-aa] quit
```

On DeviceB, configure a BFD session between DeviceA and DeviceB.

```
<DeviceB> system-view
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] bfd bb bind peer-ip 1.1.1.1
[*DeviceB-bfd-session-bb] discriminator local 20
[*DeviceB-bfd-session-bb] discriminator remote 10
[*DeviceB-bfd-session-bb] commit
[~DeviceB-bfd-session-bb] quit
```

Step 3 Configure a default static route and bind it to a BFD session.

On DeviceA, configure a default static route to the external network and bind it to BFD session named **aa**.

```
[~DeviceA] ip route-static 0.0.0.0 0 1.1.1.2 track bfd-session aa
```

Step 4 Verify the configuration.

Run the **display bfd session all** command on DeviceA and DeviceB. The command output shows that a BFD session has been established and is up. Then, run the **display current-configuration | include bfd** command in the system view. The command output shows that the static route has been bound to the BFD session.

Use the command output on DeviceA as an example.

```
[~DeviceA] display bfd session all
-----
Local  Remote PeerIpAddr   State    Type      InterfaceName
-----
10    20    1.1.1.2        Up      S_IP_PEER  -
-----
Total UP/DOWN Session Number : 1/0
[~DeviceA] display current-configuration | include bfd
```

```
bfd
bfd aa bind peer-ip 1.1.1.2
ip route-static 0.0.0.0 0.0.0.0 1.1.1.2 track bfd-session aa
```

Check the IP routing table of DeviceA. The command output shows that the static route exists in the routing table.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table: Public
Destinations : 5      Routes : 5
Destination/Mask Proto Pre Cost   Flags NextHop     Interface
0.0.0.0/0    Static 60 0        RD 1.1.1.2      GigabitEthernet1/0/0
1.1.1.0/24   Direct 0 0        D  1.1.1.1      GigabitEthernet1/0/0
1.1.1.1/32   Direct 0 0        D  127.0.0.1    GigabitEthernet1/0/0
1.1.1.255/32 Direct 0 0        D  127.0.0.1    GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0        D  127.0.0.1    InLoopBack0
```

Run the **shutdown** command on GE 1/0/0 of DeviceB to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] shutdown
```

Check the IP routing table of DeviceA. The command output shows that the static default route 0.0.0.0/0 does not exist. This is because the default static route has become unavailable after the BFD session bound to the route detects the link fault.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table: Public
Destinations : 4      Routes : 4
Destination/Mask Proto Pre Cost   Flags NextHop     Interface
1.1.1.0/24   Direct 0 0        D  1.1.1.1      GigabitEthernet1/0/0
1.1.1.1/32   Direct 0 0        D  127.0.0.1    GigabitEthernet1/0/0
1.1.1.255/32 Direct 0 0        D  127.0.0.1    GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0        D  127.0.0.1    InLoopBack0
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 1.1.1.1 255.255.255.0
#
ip route-static 0.0.0.0 0.0.0.0 1.1.1.2 track bfd-session aa
#
bfd aa bind peer-ip 1.1.1.2
discriminator local 10
discriminator remote 20
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
```

```
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 2.2.2.2 255.255.255.0
#
bfd bb bind peer-ip 1.1.1.1
discriminator local 20
discriminator remote 10
#
return
```

Example for Configuring NQA for IPv4 Static Routes

NQA for IPv4 static routes can fast detect network faults and control the advertisement of static routes.

Networking Requirements

On a simple network or when the router cannot use a dynamic routing protocol to generate routes, you can configure static routes. Unlike dynamic routing protocols, static routes do not have a detection mechanism. If a link fails, a network administrator must manually delete the corresponding static route from the IP routing table, which delays link switchovers and causes a lengthy service interruption.

Bidirectional Forwarding Detection (BFD) for static routes is adaptable to link changes but requires that both ends of a link support BFD. If either end of a link does not support BFD, configure NQA for IPv4 static routes. If an NQA test instance detects a link fault, it instructs the routing management module to delete the associated static route from the IPv4 routing table. Then traffic is switched to a backup route to prevent lengthy service interruptions.

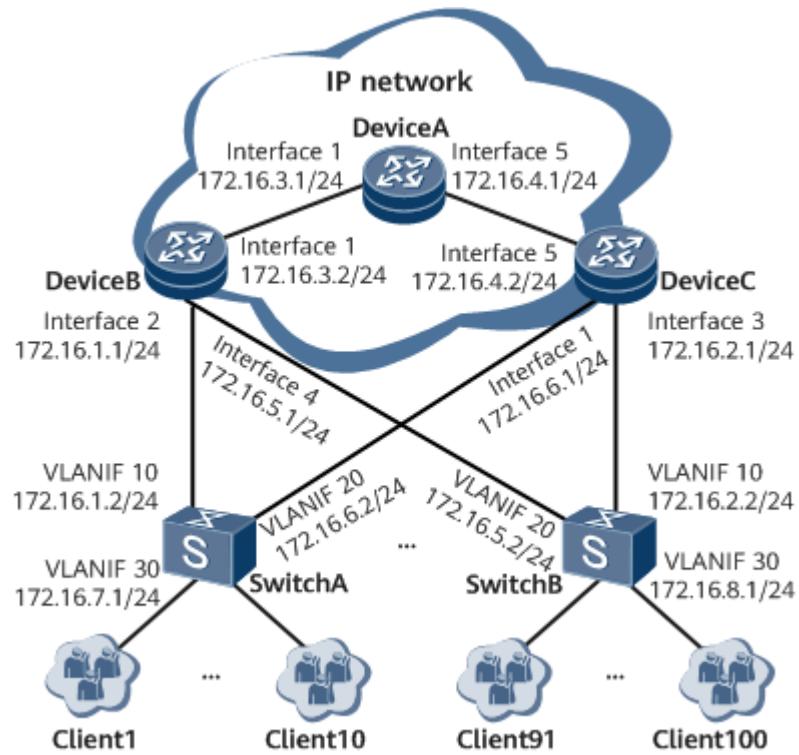
In [Figure 1-46](#), backup links are deployed on the IP metropolitan area network (MAN).

- Static routes are configured on DeviceB and DeviceC. DeviceB is the active device, while DeviceC is the standby device.
- In most cases, traffic is transmitted over the primary link (DeviceB -> SwitchA).
- If the primary link fails, traffic switches to the backup link (DeviceC -> SwitchA).

Figure 1-46 NQA for IPv4 static routes

NOTE

Interfaces 1 through 5 in this example represent GE 1/0/0, GE 1/0/1, GE 1/0/2, GE 1/0/3, and GE 2/0/3, respectively.



NOTE

In this example, switches A and B are used for user access. In actual networking, users can also access the network through the OLT, DSLAM, MSAN, or xDSL. The configurations on DeviceA, DeviceB, and DeviceC are the same.

Configuration Roadmap

The configuration roadmap is as follows:

1. Create an Internet Control Message Protocol (ICMP) NQA test instance to monitor the status of the primary link.
Create an ICMP NQA test instance on the NQA client DeviceB to test whether the primary link is running properly.
2. Configure static routes and associate the static route along the primary link with the ICMP NQA test instance.
Configure static routes on DeviceB and DeviceC, and associate the static route configured on DeviceB with the ICMP NQA test instance. If the ICMP NQA test instance detects a link fault, it instructs the routing management module to delete the associated static route from the IPv4 routing table.
3. Configure a dynamic routing protocol.
Configure a dynamic routing protocol on DeviceA, DeviceB, and DeviceC so that they can learn routes from one another.
4. Configure OSPF to import static routes and set a smaller cost for the static route along the primary link than for the one along the backup link.
Configure OSPF on DeviceB and DeviceC to import static routes, and set a higher cost for the static route imported by DeviceC than for the one

imported by DeviceB. This configuration allows DeviceA to preferentially select the link (DeviceB -> SwitchA).

Data Preparation

To complete the configuration, you need the following data:

- IP address of each interface
- NQA item values (for details, see [Table 1-11](#))

Table 1-11 NQA item values

Item	Value
Administrator name	user
Name of the test instance	test
Test type	ICMP
Destination address	172.16.1.2
Interval at which the NQA test automatically runs	10 seconds
Number of probes	2
Interval at which probe packets are sent	5 seconds
Timeout period	4 seconds

- OSPF backbone area (Area 0) of DeviceA, DeviceB, and DeviceC, and their router IDs (1.1.1.1, 2.2.2.2, and 3.3.3.3)

Procedure

Step 1 Configure interface IPv4 addresses. For configuration details, see [Configuration Files](#) in this section.

Step 2 Create an NQA test instance on DeviceB to test the link between DeviceB and SwitchA.

```
<DeviceB> system-view
[~DeviceB] nqa test-instance user test
[*DeviceB-nqa-user-test] test-type icmp
[*DeviceB-nqa-user-test] destination-address ipv4 172.16.1.2
[*DeviceB-nqa-user-test] frequency 10
[*DeviceB-nqa-user-test] probe-count 2
[*DeviceB-nqa-user-test] interval seconds 5
[*DeviceB-nqa-user-test] timeout 4
[*DeviceB-nqa-user-test] start now
[*DeviceB-nqa-user-test] commit
[~DeviceB-nqa-user-test] quit
```

Step 3 Configure IPv4 static routes.

```
# Configure an IPv4 static route on DeviceB and associate it with the NQA test instance.
```

```
[~DeviceB] ip route-static 172.16.7.0 255.255.255.0 GigabitEthernet 1/0/1 172.16.1.2 track nqa user test  
[*DeviceB] commit
```

Configure an IPv4 static route on DeviceC.

```
[~DeviceC] ip route-static 172.16.7.0 255.255.255.0 GigabitEthernet 1/0/0 172.16.6.2  
[*DeviceC] commit
```

Step 4 Configure a dynamic routing protocol on DeviceA, DeviceB, and DeviceC. OSPF is used in this example.

Configure OSPF on DeviceA.

```
[~DeviceA] ospf 1  
[*DeviceA-ospf-1] area 0.0.0.0  
[*DeviceA-ospf-1-area-0.0.0.0] network 172.16.3.0 0.0.0.255  
[*DeviceA-ospf-1-area-0.0.0.0] network 172.16.4.0 0.0.0.255  
[*DeviceA-ospf-1-area-0.0.0.0] quit  
[*DeviceA-ospf-1] quit  
[*DeviceA] commit
```

Configure OSPF on DeviceB.

```
[~DeviceB] ospf 1  
[*DeviceB-ospf-1] area 0.0.0.0  
[*DeviceB-ospf-1-area-0.0.0.0] network 172.16.3.0 0.0.0.255  
[*DeviceB-ospf-1-area-0.0.0.0] quit  
[*DeviceB-ospf-1] quit  
[*DeviceB] commit
```

Configure OSPF on DeviceC.

```
[~DeviceC] ospf 1  
[*DeviceC-ospf-1] area 0.0.0.0  
[*DeviceC-ospf-1-area-0.0.0.0] network 172.16.4.0 0.0.0.255  
[*DeviceC-ospf-1-area-0.0.0.0] quit  
[*DeviceC-ospf-1] quit  
[*DeviceC] commit
```

Step 5 Configure OSPF on DeviceB and DeviceC to import static routes.

Configure OSPF on DeviceB to import a static route, and set the cost to 10 for the static route.

```
[~DeviceB] ospf 1  
[*DeviceB-ospf-1] import-route static cost 10  
[*DeviceB-ospf-1] commit  
[*DeviceB-ospf-1] quit
```

Configure OSPF on DeviceC to import a static route, and set the cost to 20 for the static route.

```
[~DeviceC] ospf 1  
[*DeviceC-ospf-1] import-route static cost 20  
[*DeviceC-ospf-1] commit  
[*DeviceC-ospf-1] quit
```

Step 6 Verify the configuration.

After the configuration is complete, run the **display current-configuration | include nqa** command on DeviceB in the system view. The command output shows that the IPv4 static route has been associated with the NQA test instance. Run the **display nqa results** command. The command output shows that an NQA test instance has been created.

Display configurations of NQA for IPv4 static routes.

```
[~DeviceB] display current-configuration | include nqa
```

```
ip route-static 172.16.7.0 255.255.255.0 GigabitEthernet 1/0/1 172.16.1.2 track nqa user test  
nqa test-instance user test
```

Display NQA test results.

```
[~DeviceB] display nqa results test-instance user test  
NQA entry(user, test) : testflag is active ,testtype is icmp  
1 . Test 6645 result The test is finished  
Send operation times: 2 Receive response times: 2  
Completion:success RTD OverThresholds number:0  
Attempts number:1 Drop operation number:0  
Disconnect operation number:0 Operation timeout number:0  
System busy operation number:0 Connection fail number:0  
Operation sequence errors number:0 RTT Stats errors number:0  
Destination ip address:172.16.1.2  
Min/Max/Average Completion Time: 1/1/1  
Sum/Square-Sum Completion Time: 2/2  
Last Good Probe Time: 2012-11-14 04:20:36.9  
Lost packet ratio: 0 %
```

The command output shows "Lost packet ratio 0 %," indicating that the link is running properly.

Display the routing table on DeviceB.

```
[~DeviceB] display ip routing-table  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route  
-----  
Routing Table : _public_  
Destinations : 15 Routes : 15  
  
Destination/Mask Proto Pre Cost Flags NextHop Interface  
  
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0  
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0  
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0  
172.16.1.0/24 Direct 0 0 D 172.16.1.1 GigabitEthernet1/0/1  
172.16.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/1  
172.16.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/1  
172.16.3.0/24 Direct 0 0 D 172.16.3.2 GigabitEthernet1/0/0  
172.16.3.2/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0  
172.16.3.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0  
172.16.4.0/24 OSPF 10 2 D 172.16.3.1 GigabitEthernet1/0/0  
172.16.5.0/24 Direct 0 0 D 172.16.5.1 GigabitEthernet1/0/3  
172.16.5.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/3  
172.16.5.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/3  
172.16.7.0/24 Static 60 0 D 172.16.1.2 GigabitEthernet1/0/1  
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

The command output shows that the static route exists in the routing table.

Display the routing table on DeviceA.

```
[~DeviceA] display ip routing-table  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route  
-----  
Routing Table : _public_  
Destinations : 11 Routes : 11  
  
Destination/Mask Proto Pre Cost Flags NextHop Interface  
  
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0  
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0  
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0  
172.16.3.0/24 Direct 0 0 D 172.16.3.1 GigabitEthernet1/0/0  
172.16.3.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0  
172.16.3.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0  
172.16.4.0/24 Direct 0 0 D 172.16.4.1 GigabitEthernet2/0/3  
172.16.4.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/3
```

```
172.16.4.255/32 Direct 0 0      D 127.0.0.1    GigabitEthernet2/0/3
  172.16.7.0/24 O_ASE 150 10    D 172.16.3.2    GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0      D 127.0.0.1    InLoopBack0
```

The command output shows that a route to 172.16.7.0/24 exists in the routing table. The route's next hop address is 172.16.3.2 and the cost is 10. Traffic is preferentially transmitted along the link DeviceB -> SwitchA.

Shut down GE 1/0/1 on DeviceB to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/1
[~DeviceB-GigabitEthernet1/0/1] shutdown
[~DeviceB-GigabitEthernet1/0/1] commit
[~DeviceB] quit
```

Display NQA test results.

```
[~DeviceB] display nqa results test-instance user test
NQA entry(user, test) : testflag is active ,testtype is icmp
1 . Test 7160 result  The test is finished
Send operation times: 2           Receive response times: 0
Completion:failed           RTD OverThresholds number:0
Attempts number:1           Drop operation number:0
Disconnect operation number:0     Operation timeout number:2
System busy operation number:0     Connection fail number:0
Operation sequence errors number:0   RTT Stats errors number:0
Destination ip address:172.16.1.2
Min/Max/Average Completion Time: 0/0/0
Sum/Square-Sum Completion Time: 0/0
Last Good Probe Time: 0000-00-00 00:00:00.0
Lost packet ratio: 100 %
```

The command output shows "Completion:failed" and "Lost packet ratio is 100 %," indicating that the link is faulty.

Display the routing table on DeviceB.

```
[~DeviceB] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 12      Routes : 12
Destination/Mask Proto Pre Cost      Flags NextHop      Interface
127.0.0.0/8 Direct 0 0      D 127.0.0.1    InLoopBack0
127.0.0.1/32 Direct 0 0      D 127.0.0.1    InLoopBack0
127.255.255.255/32 Direct 0 0      D 127.0.0.1    InLoopBack0
  172.16.3.0/24 Direct 0 0      D 172.16.3.2    GigabitEthernet1/0/0
  172.16.3.2/32 Direct 0 0      D 127.0.0.1    GigabitEthernet1/0/0
  172.16.3.255/32 Direct 0 0      D 127.0.0.1    GigabitEthernet1/0/0
  172.16.4.0/24 OSPF 10 2      D 172.16.3.1    GigabitEthernet1/0/0
  172.16.5.0/24 Direct 0 0      D 172.16.5.1    GigabitEthernet1/0/3
  172.16.5.1/32 Direct 0 0      D 127.0.0.1    GigabitEthernet1/0/3
  172.16.5.255/32 Direct 0 0      D 127.0.0.1    GigabitEthernet1/0/3
  172.16.7.0/24 O_ASE 150 20    D 172.16.3.1    GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0      D 127.0.0.1    InLoopBack0
```

The command output shows that the static route has been deleted.

Display the routing table on DeviceA.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 11      Routes : 11
Destination/Mask Proto Pre Cost      Flags NextHop      Interface
```

```
127.0.0.0/8 Direct 0 0      D 127.0.0.1    InLoopBack0
127.0.0.1/32 Direct 0 0     D 127.0.0.1    InLoopBack0
127.255.255.255/32 Direct 0 0   D 127.0.0.1    InLoopBack0
172.16.3.0/24 Direct 0 0     D 172.16.3.1    GigabitEthernet1/0/0
172.16.3.1/32 Direct 0 0     D 127.0.0.1    GigabitEthernet1/0/0
172.16.3.255/32 Direct 0 0   D 127.0.0.1    GigabitEthernet1/0/0
172.16.4.0/24 Direct 0 0     D 172.16.4.1    GigabitEthernet2/0/3
172.16.4.1/32 Direct 0 0     D 127.0.0.1    GigabitEthernet2/0/3
172.16.4.255/32 Direct 0 0   D 127.0.0.1    GigabitEthernet2/0/3
172.16.7.0/24 O_ASE 150 20   D 172.16.4.2    GigabitEthernet2/0/3
255.255.255.255/32 Direct 0 0   D 127.0.0.1    InLoopBack0
```

The static route has been associated with the NQA test instance on DeviceB. If NQA detects a link fault, it rapidly notifies DeviceB that the associated static route is unavailable. DeviceA cannot learn the route to 172.16.7.0/24 from DeviceB. However, DeviceA can learn the route to 172.16.7.0/24 from DeviceC. The route's next hop address is 172.16.4.2, and the cost is 20. Traffic switches to the link DeviceC -> SwitchA.

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA  
#  
router id 1.1.1.1  
#  
interface GigabitEthernet 1/0/0  
undo shutdown  
ip address 172.16.3.1 255.255.255.0  
#  
interface GigabitEthernet 2/0/3  
undo shutdown  
ip address 172.16.4.1 255.255.255.0  
#  
ospf 1  
area 0.0.0.0  
network 172.16.3.0 0.0.0.255  
network 172.16.4.0 0.0.0.255  
#  
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
router id 2.2.2.2  
#  
interface GigabitEthernet 1/0/0  
undo shutdown  
ip address 172.16.3.2 255.255.255.0  
#  
interface GigabitEthernet 1/0/1  
undo shutdown  
ip address 172.16.1.1 255.255.255.0  
#  
interface GigabitEthernet 1/0/3  
undo shutdown  
ip address 172.16.5.1 255.255.255.0  
#  
ospf 1  
import-route static cost 10  
area 0.0.0.0
```

```
network 172.16.3.0 0.0.0.255
#
ip route-static 172.16.7.0 255.255.255.0 GigabitEthernet1/0/1 172.16.1.2 track nqa user test
#
nqa test-instance user test
test-type icmp
destination-address ipv4 172.16.1.2
interval seconds 5
timeout 4
probe-count 2
frequency 10
start now
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
interface GigabitEthernet 1/0/0
undo shutdown
ip address 172.16.6.1 255.255.255.0
#
interface GigabitEthernet 1/0/2
undo shutdown
ip address 172.16.2.1 255.255.255.0
#
interface GigabitEthernet 2/0/3
undo shutdown
ip address 172.16.4.2 255.255.255.0
#
ospf 1
import-route static cost 20
area 0.0.0
network 172.16.4.0 0.0.0.255
#
ip route-static 172.16.7.0 255.255.255.0 GigabitEthernet1/0/0 172.16.6.2
#
return
```

Example for Configuring NQA Group for IPv4 Static Route

NQA group for IPv4 static route allows an NQA group to be associated with multiple NQA test instances to monitor multiple links, rapidly detect network faults, and control advertisement of IPv4 static routes, thereby implementing service switching.

Networking Requirements

NQA group for IPv4 static route monitors the link status of an IPv4 static route through association between the static route and an NQA group that consists of multiple NQA test instances. The RM module determines whether the static route is active based on the test result of the NQA group. If the IPv4 static route is inactive, the RM module deletes it from the IP routing table and selects an available backup link for data forwarding, which prevents lengthy service interruptions.

[Figure 1-47](#) shows a network with redundant links.

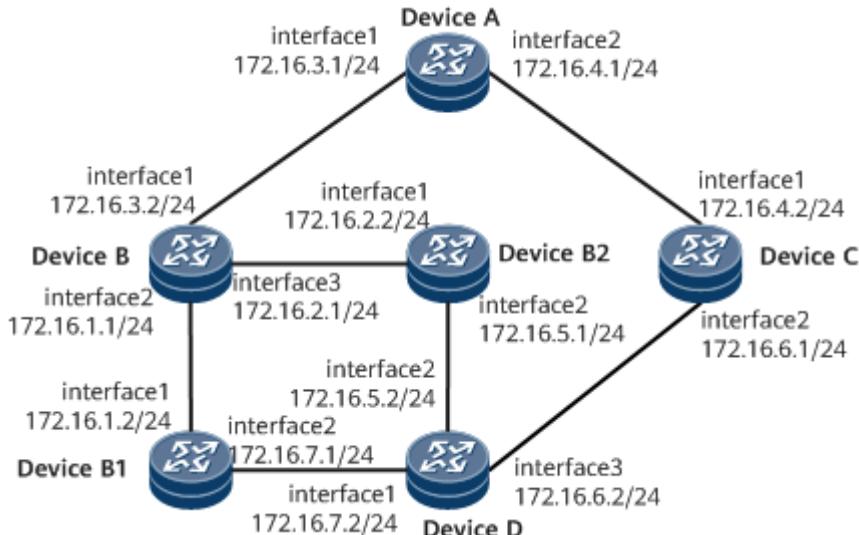
- BGP peer relationships are established between Device A and Device B and between Device A and Device C.

- On Device B and Device C, default static routes are configured and imported into BGP, and different PrefVal values are configured so that Device B and Device C function as the master device and backup device, respectively.
- Device B and Device D are connected through two links by way of Device B1 and Device B2.
- In normal cases, traffic is transmitted along the primary link (Device B -> Device D).
- If the primary link fails (the NQA group bound to the default route of Device B considers the IPv4 static route inactive when monitoring the two links from Device B to Device D), traffic is switched to the backup link (Device C -> Device D).

Figure 1-47 Network diagram of NQA group for IPv4 static route

 NOTE

In this example, interface1, interface2, and interface3 represent GE 1/0/0, GE 1/0/1, and GE 1/0/2, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Create NQA test instances of the ICMP type to detect faults on the primary link.
Create two NQA test instances of the ICMP type between Device B and Device B1 and between Device B and Device B2 to correspond to the two links between Device B and Device D and to check whether the primary link (Device B -> Device D) is running properly.
2. Create an NQA group.
Bind the two NQA test instances of the ICMP type to the NQA group.
3. Configure static routes and associate the static route along the primary link with the NQA group.
Configure IPv4 static routes to loopback0 interfaces between Device A and Device B, and between Device A and Device C. Configure default IPv4 static

routes on Device B and Device C. Associate the IPv4 static route configured on Device B with the NQA group, which collects the detection results of two NQA test instances to determine its status. When the NQA group goes down, the device determines that the current link is faulty and instructs the RM module to delete the IPv4 static route from the IP routing table.

4. Configure BGP.

Configure BGP on Device A, Device B, and Device C so that they can learn routes from each other.

5. Configure BGP to import IPv4 static routes and set a larger PrefVal value for the primary link.

Configure BGP on Device B and Device C to import IPv4 static routes, and set a larger PrefVal value for the IPv4 static route imported by Device B. When Device A learns routes to the same destination from Device B and Device C, it preferentially selects the link (Device B -> Device D), which has a larger PrefVal value.

Data Preparation

To complete the configuration, you need the following data:

- IP address of each interface
- NQA-related configurations (for details, see [Table 1-12](#))

Table 1-12 NQA parameter values

Item	Value
Administrator name	user1, user2
Test instance name	test1, test2
Test type	ICMP
Destination address	172.16.1.2, 172.16.2.2
Test interval	10s
Number of probes	2
Interval at which packets are sent	5s
Timeout period	4s

- Operation type between test instances in the NQA group: OR
- BGP AS numbers (the same) and router IDs of Device A, Device B, and Device C (1.1.1.1, 2.2.2.2, and 3.3.3.3, respectively)

Procedure

Step 1 Configure IP addresses. For details, see [Configuration Files](#).

Step 2 On Device B, configure two NQA test instances of the ICMP type between Device B and Device B1 and between Device B and Device B2.

```
<DeviceB> system-view
[~DeviceB] nqa test-instance user1 test1
[*DeviceB-nqa-user1-test1] test-type icmp
[*DeviceB-nqa-user1-test1] destination-address ipv4 172.16.1.2
[*DeviceB-nqa-user1-test1] frequency 10
[*DeviceB-nqa-user1-test1] probe-count 2
[*DeviceB-nqa-user1-test1] interval seconds 5
[*DeviceB-nqa-user1-test1] timeout 4
[*DeviceB-nqa-user1-test1] start now
[*DeviceB-nqa-user1-test1] commit
[~DeviceB-nqa-user1-test1] quit
[~DeviceB] nqa test-instance user2 test2
[*DeviceB-nqa-user2-test2] test-type icmp
[*DeviceB-nqa-user1-test1] destination-address ipv4 172.16.2.2
[*DeviceB-nqa-user2-test2] frequency 10
[*DeviceB-nqa-user2-test2] probe-count 2
[*DeviceB-nqa-user2-test2] interval seconds 5
[*DeviceB-nqa-user2-test2] timeout 4
[*DeviceB-nqa-user2-test2] start now
[*DeviceB-nqa-user2-test2] commit
[~DeviceB-nqa-user2-test2] quit
```

Step 3 Create an NQA group and bind the two NQA test instances of the ICMP type to the group.

```
[~DeviceB] nqa group group1
[*DeviceB-nqa-group-group1] nqa test-instance user1 test1
[*DeviceB-nqa-group-group1] nqa test-instance user2 test2
[*DeviceB-nqa-group-group1] operator or
[*DeviceB-nqa-group-group1] commit
[~DeviceB-nqa-group-group1] quit
```

Step 4 Configure static routes.

Configure static routes to loopback interfaces between Device A and Device B, and between Device A and Device C.

```
[~DeviceA] ip route-static 2.2.2.2 255.255.255.255 GigabitEthernet 1/0/0 172.16.3.2
[~DeviceA] ip route-static 3.3.3.3 255.255.255.255 GigabitEthernet 1/0/1 172.16.4.2
[*DeviceA] commit
```

Configure a static route to the loopback interface of Device A on Device B.

```
[~DeviceB] ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet 1/0/0 172.16.3.1
[*DeviceB] commit
```

Configure a static route to the loopback interface of Device A on Device C.

```
[~DeviceC] ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet 1/0/1 172.16.4.1
[*DeviceC] commit
```

Configure a default static route on Device B and associate it with the NQA group.

```
[*DeviceB] ip route-static 0.0.0.0 32 NULL0 track nqa-group group1
[*DeviceB] commit
```

Configure a default static route on Device C.

```
[*DeviceC] ip route-static 0.0.0.0 32 NULL0
[*DeviceC] commit
```

Step 5 Configure BGP on Device A, Device B, and Device C.

Configure BGP on Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 2.2.2.2 as-number 100
[*DeviceA-bgp] peer 2.2.2.2 connect-interface Loopback0
```

```
[*DeviceA-bgp] peer 3.3.3.3 as-number 100
[*DeviceA-bgp] peer 3.3.3.3 connect-interface Loopback0
[*DeviceA-bgp] quit
[*DeviceA] commit
```

Configure BGP on Device B.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] peer 1.1.1.1 as-number 100
[*DeviceB-bgp] peer 1.1.1.1 connect-interface Loopback0
[*DeviceB-bgp] quit
[*DeviceB] commit
```

Configure BGP on Device C.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] peer 1.1.1.1 as-number 100
[*DeviceC-bgp] peer 1.1.1.1 connect-interface Loopback0
[*DeviceC-bgp] quit
[*DeviceC] commit
```

Step 6 Configure BGP on Device B and Device C to import IPv4 static routes, and set a larger PrefVal value for the primary link.

Configure BGP on Device B to import IPv4 static routes, and set the PrefVal value to 200.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] import-route static
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] peer 1.1.1.1 preferred-value 200
[*DeviceB-bgp-af-ipv4] commit
[*DeviceB-bgp-af-ipv4] quit
[*DeviceB-bgp] quit
```

Configure BGP on Device C to import IPv4 static routes, and set the PrefVal value to 100.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] import-route static
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] peer 1.1.1.1 preferred-value 100
[*DeviceC-bgp-af-ipv4] commit
[*DeviceC-bgp-af-ipv4] quit
[*DeviceC-bgp] quit
```

Step 7 Verify the configuration.

Check the results of the two NQA test instances on Device B.

```
[~DeviceB] display nqa results test-instance user1 test1
NQA entry(user1, test1) :testflag is inactive ,testtype is icmp
1 . Test 1 result  The test is finished
Send operation times: 3          Receive response times: 3
Completion:success           RTD OverThresholds number:0
Attempts number:1            Drop operation number:0
Disconnect operation number:0    Operation timeout number:0
System busy operation number:0   Connection fail number:0
Operation sequence errors number:0   RTT Status errors number:0
Destination ip address:172.16.1.2
Min/Max/Average Completion Time: 11/185/69
Sum/Square-Sum Completion Time: 207/34467
Last Good Probe Time: 2022-09-24 15:08:38.6
Lost packet ratio: 0 %

[~DeviceB] display nqa results test-instance user2 test2
NQA entry(user2, test2) :testflag is inactive ,testtype is icmp
1 . Test 1 result  The test is finished
Send operation times: 3          Receive response times: 3
Completion:success           RTD OverThresholds number:0
```

```
Attempts number:1          Drop operation number:0
Disconnect operation number:0   Operation timeout number:0
System busy operation number:0   Connection fail number:0
Operation sequence errors number:0   RTT Status errors number:0
Destination ip address:172.16.2.2
Min/Max/Average Completion Time: 9/16/11
Sum/Square-Sum Completion Time: 34/418
Last Good Probe Time: 2022-09-24 15:08:50.3
Lost packet ratio: 0 %
```

The command output shows "Lost packet ratio 0 %," indicating that the link is running properly.

Check the test result of the NQA group on Device B.

```
[~DeviceB] display nqa group
NQA-group information:
-----
NQA-group group1
Status: UP      Operator: OR
-----
Admin-name           Test-name       Status
-----
user1               test1          UP
user2               test2          UP
```

Check the routing table of Device A.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 14      Routes : 14
-----
Destination/Mask  Proto  Pre  Cost    Flags NextHop          Interface
0.0.0.0/32        IBGP   255  0        RD  2.2.2.2        GigabitEthernet1/0/0
1.1.1.1/32        Direct 0  0        D   127.0.0.1       LoopBack0
2.2.2.2/32        Static  60  0        D   172.16.3.2      GigabitEthernet1/0/0
3.3.3.3/32        Static  60  0        D   172.16.4.2      GigabitEthernet1/0/1
127.0.0.0/8       Direct  0  0        D   127.0.0.1       InLoopBack0
127.0.0.1/32      Direct  0  0        D   127.0.0.1       InLoopBack0
127.255.255.255/32 Direct 0  0        D   127.0.0.1       InLoopBack0
172.16.3.0/24     Direct 0  0        D   172.16.3.1       GigabitEthernet1/0/0
172.16.3.1/32     Direct 0  0        D   127.0.0.1       GigabitEthernet1/0/0
172.16.3.255/32   Direct 0  0        D   127.0.0.1       GigabitEthernet1/0/0
172.16.4.0/24     Direct 0  0        D   172.16.4.1       GigabitEthernet1/0/1
172.16.4.1/32     Direct 0  0        D   127.0.0.1       GigabitEthernet1/0/1
172.16.4.255/32   Direct 0  0        D   127.0.0.1       GigabitEthernet1/0/1
255.255.255.255/32 Direct 0  0        D   127.0.0.1       InLoopBack0
```

The command output shows that the next hop used to forward user traffic on Device A is Device B, indicating that user traffic is transmitted along the primary link.

Shut down GigabitEthernet 1/0/1 on Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/1
[~DeviceB-GigabitEthernet1/0/1] shutdown
[~DeviceB-GigabitEthernet1/0/1] commit
[~DeviceB] quit
```

Check the NQA test result.

```
[~DeviceB] display nqa results test-instance user1 test1
NQA entry(user1, test1) :testflag is inactive ,testtype is icmp
2 . Test 2 result  The test is finished
Send operation times: 3          Receive response times: 0
Completion:failed          RTD OverThresholds number:0
```

```
Attempts number:1          Drop operation number:0
Disconnect operation number:0   Operation timeout number:3
System busy operation number:0   Connection fail number:0
Operation sequence errors number:0   RTT Status errors number:0
Destination ip address:172.16.1.2
Min/Max/Average Completion Time: 0/0/0
Sum/Square-Sum Completion Time: 0/0
Last Good Probe Time: 0000-00-00 00:00:00.0
Lost packet ratio: 100 %
```

"Completion:failed" and "Lost packet ratio: 100%" are displayed, indicating that the link where GigabitEthernet 1/0/1 resides is faulty.

Check the test result of the NQA group on Device B.

```
[~DeviceB] display nqa group
NQA-group information:
-----
NQA-group group1
Status: UP      Operator: OR
-----
Admin-name           Test-name        Status
-----
user1               test1           DOWN
user2               test2           UP
```

The operation type between test instances in the NQA group is OR. Therefore, the status of the NQA group remains UP.

Check the routing table of Device A.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 14      Routes : 14
-----
Destination/Mask  Proto  Pre  Cost    Flags NextHop          Interface
0.0.0.0/32  IBGP  255  0        RD 2.2.2.2          GigabitEthernet1/0/0
1.1.1.1/32  Direct 0  0        D  127.0.0.1        LoopBack0
2.2.2.2/32  Static 60 0       D  172.16.3.2        GigabitEthernet1/0/0
3.3.3.3/32  Static 60 0       D  172.16.4.2        GigabitEthernet1/0/1
127.0.0.0/8  Direct 0  0        D  127.0.0.1        InLoopBack0
127.0.0.1/32 Direct 0  0        D  127.0.0.1        InLoopBack0
127.255.255.255/32 Direct 0  0       D  127.0.0.1        InLoopBack0
172.16.3.0/24 Direct 0  0        D  172.16.3.1        GigabitEthernet1/0/0
172.16.3.1/32 Direct 0  0        D  127.0.0.1        GigabitEthernet1/0/0
172.16.3.255/32 Direct 0  0       D  127.0.0.1        GigabitEthernet1/0/0
172.16.4.0/24 Direct 0  0        D  172.16.4.1        GigabitEthernet1/0/1
172.16.4.1/32 Direct 0  0        D  127.0.0.1        GigabitEthernet1/0/1
172.16.4.255/32 Direct 0  0       D  127.0.0.1        GigabitEthernet1/0/1
255.255.255.255/32 Direct 0  0       D  127.0.0.1        InLoopBack0
```

The command output shows that the next hop used to forward user traffic on Device A is still Device B, indicating that user traffic is transmitted along the primary link.

Shut down GigabitEthernet 1/0/2 on Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/2
[~DeviceB-GigabitEthernet1/0/2] shutdown
[*DeviceB-GigabitEthernet1/0/2] commit
[~DeviceB] quit
```

Check the NQA test result.

```
[~DeviceB] display nqa results test-instance user1 test1
```

```
NQA entry(user2, test2) : testflag is active ,testtype is icmp
1 . Test 186 result  The test is finished
Send operation times: 2          Receive response times: 0
Completion:failed           RTD OverThresholds number:0
Attempts number:1            Drop operation number:0
Disconnect operation number:0    Operation timeout number:2
System busy operation number:0   Connection fail number:0
Operation sequence errors number:0   RTT Stats errors number:0
Destination ip address:172.16.2.2
Min/Max/Average Completion Time: 0/0/0
Sum/Square-Sum Completion Time: 0/0
Last Good Probe Time: 0000-00-00 00:00:00.0
Lost packet ratio: 100 %
```

"Completion:failed" and "Lost packet ratio: 100%" are displayed, indicating that the link where GigabitEthernet 1/0/2 resides is faulty.

Check the test result of the NQA group on Device B.

```
[~DeviceB] display nqa group
NQA-group information:
-----
NQA-group group1
Status: DOWN      Operator: OR
-----
Admin-name        Test-name        Status
-----
user1            test1            DOWN
user2            test2            DOWN
```

The command output shows that the status of the NQA group changes to DOWN.

Check the routing table of Device A.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 14      Routes : 14
-----
Destination/Mask Proto Pre Cost      Flags NextHop          Interface
0.0.0.0/32     IBGP 255 0        RD 3.3.3.3          GigabitEthernet1/0/1
1.1.1.1/32     Direct 0 0        D 127.0.0.1         LoopBack0
2.2.2.2/32     Static 60 0       D 172.16.3.2         GigabitEthernet1/0/0
3.3.3.3/32     Static 60 0       D 172.16.4.2         GigabitEthernet1/0/1
127.0.0.0/8    Direct 0 0        D 127.0.0.1         InLoopBack0
127.0.0.1/32   Direct 0 0        D 127.0.0.1         InLoopBack0
127.255.255.255/32 Direct 0 0       D 127.0.0.1         InLoopBack0
172.16.3.0/24   Direct 0 0        D 172.16.3.1         GigabitEthernet1/0/0
172.16.3.1/32   Direct 0 0        D 127.0.0.1         GigabitEthernet1/0/0
172.16.3.255/32 Direct 0 0       D 127.0.0.1         GigabitEthernet1/0/0
172.16.4.0/24   Direct 0 0        D 172.16.4.1         GigabitEthernet1/0/1
172.16.4.1/32   Direct 0 0        D 127.0.0.1         GigabitEthernet1/0/1
172.16.4.255/32 Direct 0 0       D 127.0.0.1         GigabitEthernet1/0/1
255.255.255.255/32 Direct 0 0       D 127.0.0.1         InLoopBack0
```

The command output shows that the next hop used to forward user traffic on Device A changes to Device C, indicating that the traffic is switched to the backup link.

The NQA group on Device B is associated with the IPv4 static route. When the NQA group detects that both links from Device B to Device D fail, the NQA group goes down and rapidly notifies Device B that the associated IPv4 static route is unavailable. Service traffic is then switched to the backup link.

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA  
#  
router id 1.1.1.1  
#  
interface LoopBack0  
ip address 1.1.1.1 255.255.255.255  
#  
interface GigabitEthernet 1/0/0  
ip address 172.16.3.1 255.255.255.0  
#  
interface GigabitEthernet 1/0/1  
ip address 172.16.4.1 255.255.255.0  
#  
bgp 100  
peer 2.2.2.2 as-number 100  
peer 2.2.2.2 connect-interface LoopBack0  
peer 3.3.3.3 as-number 100  
peer 3.3.3.3 connect-interface LoopBack0  
#  
ipv4-family unicast  
undo synchronization  
peer 2.2.2.2 enable  
peer 3.3.3.3 enable  
#  
ip route-static 2.2.2.2 255.255.255.255 GigabitEthernet 1/0/0 172.16.3.2  
ip route-static 3.3.3.3 255.255.255.255 GigabitEthernet 1/0/1 172.16.4.2  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
router id 2.2.2.2  
#  
interface LoopBack0  
ip address 2.2.2.2 255.255.255.255  
#  
interface GigabitEthernet 1/0/0  
ip address 172.16.3.2 255.255.255.0  
#  
interface GigabitEthernet 1/0/1  
ip address 172.16.1.1 255.255.255.0  
#  
interface GigabitEthernet 1/0/2  
ip address 172.16.2.1 255.255.255.0  
#  
bgp 100  
peer 1.1.1.1 as-number 100  
peer 1.1.1.1 connect-interface LoopBack0  
#  
ipv4-family unicast  
import-route static  
peer 1.1.1.1 preferred-value 200  
#  
ip route-static 0.0.0.0 32 NULL0 track nqa-group group1  
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet 1/0/0 172.16.3.1  
#  
nqa test-instance user1 test1  
test-type icmp  
destination-address ipv4 172.16.1.2  
interval seconds 5  
timeout 4  
probe-count 2  
frequency 10
```

```
start now
#
nqa test-instance user2 test2
test-type icmp
destination-address ipv4 172.16.2.2
interval seconds 5
timeout 4
probe-count 2
frequency 10
start now
#
nqa group group1
nqa test-instance user1 test1
nqa test-instance user2 test2
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
interface LoopBack0
ip address 3.3.3.3 255.255.255.255
#
interface GigabitEthernet1/0/0
ip address 172.16.6.1 255.255.255.0
#
interface GigabitEthernet1/0/1
ip address 172.16.4.2 255.255.255.0
#
bgp 100
peer 1.1.1.1 as-number 100
peer 1.1.1.1 connect-interface LoopBack0
#
ipv4-family unicast
import-route static
peer 1.1.1.1 preferred-value 100
#
ip route-static 0.0.0.0 32 NULL0
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/1 172.16.4.1
#
return
```

Example for Associating EFM with IPv4 Static Routes

After EFM is associated with IPv4 static routes, the system responds to EFM Up/Down events on a specified interface and determines whether to activate static routes. In this manner, route advertisement is controlled and remote traffic is forwarded.

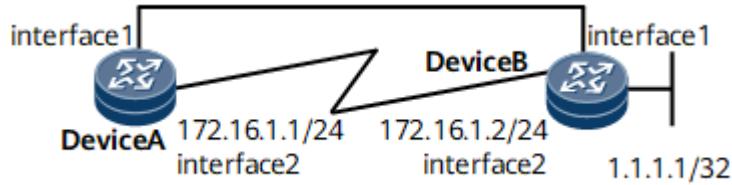
Networking Requirements

As shown in [Figure 1-48](#), DeviceA and DeviceB are connected and are enabled with EFM OAM. In addition, a static route destined for 1.1.1.1/32 is configured on DeviceA and associated with EFM.

Figure 1-48 Associating EFM with IPv4 static routes



In this example, interface1 and interface2 represent GE1/0/0 and GE1/0/1, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Enable EFM OAM both globally and on the interfaces of Device A and Device B.
2. Configure a static route destined for 1.1.1.1/32 on Device A and associate EFM with this static route.

Data Preparation

To complete the configuration, you need the IP addresses of interfaces.

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Enable EFM OAM both globally and on the interfaces of Device A and Device B.

Enable EFM OAM both globally and on the interface of Device A.

```
<DeviceA> system-view
[~DeviceA] efm enable
[*DeviceA] interface GigabitEthernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] undo shutdown
[*DeviceA-GigabitEthernet1/0/0] efm enable
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] quit
```

Enable EFM OAM both globally and on the interface of Device B.

```
<DeviceB> system-view
[~DeviceB] efm enable
[*DeviceB] interface GigabitEthernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] undo shutdown
[*DeviceB-GigabitEthernet1/0/0] efm enable
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
[~DeviceB] quit
```

Display EFM OAM session information on Device A.

```
<DeviceA> display efm session all
Interface          EFM State        Loopback Timeout
-----  
GigabitEthernet1/0/0    detect           --
```

The preceding command output shows that the EFM OAM status is **detect**.

Step 3 Associating EFM with IPv4 static routes.

Configure a static route destined for 1.1.1.1/32 on Device A and associate EFM with this static route.

```
<DeviceA> system-view
[~DeviceA] ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/1 172.16.1.2 track efm-state
GigabitEthernet1/0/0
[*DeviceA] commit
[~DeviceA] quit
```

Step 4 Verify the configuration.

Run the **display current-configuration | include efm** command on Device A. The following command output shows that the static route has been associated with EFM:

```
<DeviceA> display current-configuration | include efm
efm enable
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/1 172.16.1.2 track efm-state GigabitEthernet1/0/0
```

Display the IP routing table on Device A. The following command output shows that the static route destined for 1.1.1.1/32 exists in the IP routing table:

```
<DeviceA> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 8      Routes : 8
Destination/Mask   Proto  Pre  Cost      Flags NextHop       Interface
1.1.1.1/32  Static  60   0        D  172.16.1.2    GigabitEthernet1/0/1
127.0.0.0/8   Direct  0   0        D  127.0.0.1    InLoopBack0
127.0.0.1/32  Direct  0   0        D  127.0.0.1    InLoopBack0
127.255.255.255/32 Direct  0   0        D  127.0.0.1    InLoopBack0
172.16.1.0/24  Direct  0   0        D  172.16.1.1    GigabitEthernet1/0/1
172.16.1.1/32  Direct  0   0        D  127.0.0.1    GigabitEthernet1/0/1
172.16.1.255/32 Direct  0   0        D  127.0.0.1    GigabitEthernet1/0/1
255.255.255.255/32 Direct  0   0        D  127.0.0.1    InLoopBack0
```

Run the **shutdown** command on GE 1/0/0 of Device A to check whether the EFM OAM status changes.

```
<DeviceA> system-view
[~DeviceA] interface GigabitEthernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] shutdown
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] quit
```

Run the **display efm session all** command on Device A. The following command output shows that the EFM OAM status changes to **discovery**:

```
<DeviceA> display efm session all
Interface          EFM State        Loopback Timeout
-----
```

Interface	EFM State	Loopback Timeout
GigabitEthernet1/0/0	discovery	--

Display the IP routing table on Device A. The command output shows that the static route destined for 1.1.1.1/32 does not exist. The static route is unavailable because it has been associated with EFM and the EFM session has not been established.

```
<DeviceA> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
```

Routing Table : _public_						
Destinations : 7			Routes : 7			
Destination/Mask	Proto	Pre	Cost	Flags	NextHop	Interface
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0
127.0.0.1/32	Direct	0	0	D	127.0.0.1	InLoopBack0
127.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0
172.16.1.0/24	Direct	0	0	D	172.16.1.1	GigabitEthernet1/0/1
172.16.1.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet1/0/1
172.16.1.255/32	Direct	0	0	D	127.0.0.1	GigabitEthernet1/0/1
255.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA  
#  
efm enable  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
efm enable  
#  
interface GigabitEthernet1/0/1  
undo shutdown  
ip address 172.16.1.1 255.255.255.0  
#  
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/1 172.16.1.2 track efm-state  
GigabitEthernet1/0/0  
#  
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
efm enable  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
efm enable  
#  
interface GigabitEthernet1/0/1  
undo shutdown  
ip address 172.16.1.2 255.255.255.0  
#  
interface LoopBack0  
ip address 1.1.1.1 255.255.255.255  
#  
return
```

Example for Configuring FRR for IPv4 Static Routes on the Public Network

FRR for IPv4 static routes on the public network can fast detect link failures.

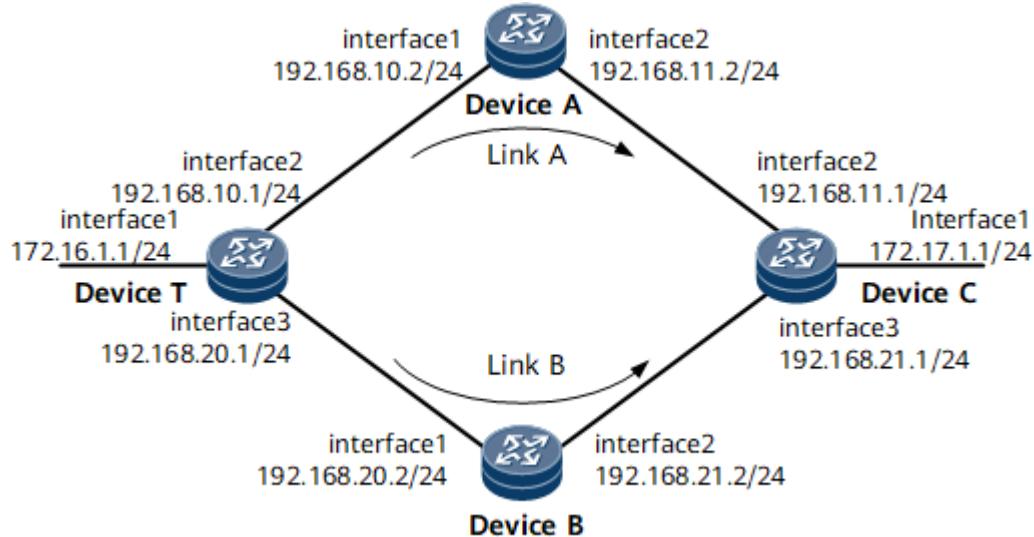
Networking Requirements

On the network shown in [Figure 1-49](#), it is required that two IPv4 static routes with Device A and Device B as the next hops be configured on Device T and that Link B function as the backup of Link A. If Link A fails, traffic is switched to Link B immediately.

Figure 1-49 Configuring FRR for IPv4 static routes on the public network

 NOTE

In this example, interface1, interface2, and interface3 represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Precautions

When configuring FRR for IPv4 static routes on the public network, ensure that there are at least two IPv4 static routes to the same destination address.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure two IPv4 static routes with Device A and Device B as the next hops on Device T.
2. Set a lower preference value for Link A on Device T so that Link A is preferentially selected.
3. Enable FRR for IPv4 static routes on Device T, and check the backup outbound interface and the backup next hop.
4. Configure static BFD for IPv4 static routes to speed up fault detection.

 NOTE

To speed up fault detection, configure dynamic or static BFD for IPv4 static routes. Static BFD for IPv4 static routes is used as an example because it is more common than dynamic BFD for IPv4 static routes on the live network.

5. Disable FRR for IPv4 static routes, and check the backup outbound interface and the backup next hop.

Data Preparation

To complete the configuration, you need preference values of IPv4 static routes.

Procedure

Step 1 Configure IP addresses for interfaces. For detailed configurations, see Configuration Files.

Step 2 Configure IPv4 static routes.

On Device A, configure IPv4 static routes.

```
[~DeviceA] ip route-static 172.16.1.0 24 GigabitEthernet1/0/0 192.168.10.1
[*DeviceA] ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.11.1
[*DeviceA] commit
```

On Device B, configure static routes.

```
[~DeviceB] ip route-static 172.16.1.0 24 GigabitEthernet1/0/0 192.168.20.1
[*DeviceB] ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.21.1
[*DeviceB] commit
```

On Device C, configure IPv4 static routes.

```
[~DeviceC] ip route-static 172.16.1.0 24 GigabitEthernet2/0/0 192.168.11.2
[*DeviceC] ip route-static 172.16.1.0 24 GigabitEthernet3/0/0 192.168.21.2
[*DeviceC] ip route-static 192.168.10.0 24 GigabitEthernet2/0/0 192.168.11.2
[*DeviceC] ip route-static 192.168.20.0 24 GigabitEthernet3/0/0 192.168.21.2
[*DeviceC] commit
```

On Device T, configure IPv4 static routes.

```
[~DeviceT] ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.10.2
[*DeviceT] ip route-static 172.17.1.0 24 GigabitEthernet3/0/0 192.168.20.2
[*DeviceT] ip route-static 192.168.11.0 24 GigabitEthernet2/0/0 192.168.10.2
[*DeviceT] ip route-static 192.168.21.0 24 GigabitEthernet3/0/0 192.168.20.2
[*DeviceT] commit
[~DeviceT] quit
```

Check the IP routing table of Device T. The following command output shows that load balancing is performed between the two IPv4 static routes.

```
<DeviceT> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 16      Routes : 16
Destination/Mask Proto Pre Cost     Flags NextHop       Interface
          127.0.0.0/8 Direct 0   0        D  127.0.0.1    InLoopBack0
          127.0.0.1/32 Direct 0   0        D  127.0.0.1    InLoopBack0
127.255.255.255/32 Direct 0   0        D  127.0.0.1    InLoopBack0
          172.16.1.0/24 Direct 0   0        D  172.16.1.1   GigabitEthernet1/0/0
          172.16.1.1/32 Direct 0   0        D  127.0.0.1   GigabitEthernet1/0/0
          172.16.1.255/32 Direct 0   0        D  127.0.0.1   GigabitEthernet1/0/0
          172.17.1.0/24 Static 60  0        D  192.168.10.2  GigabitEthernet2/0/0
                                         Static 60  0        D  192.168.20.2  GigabitEthernet3/0/0
          192.168.10.0/24 Direct 0   0        D  192.168.10.1 GigabitEthernet2/0/0
          192.168.10.1/32 Direct 0   0        D  127.0.0.1   GigabitEthernet2/0/0
          192.168.10.255/32 Direct 0   0        D  127.0.0.1   GigabitEthernet2/0/0
          192.168.11.0/24 Static 60  0        D  192.168.10.2 GigabitEthernet2/0/0
          192.168.20.0/24 Direct 0   0        D  192.168.20.1 GigabitEthernet3/0/0
          192.168.20.1/32 Direct 0   0        D  127.0.0.1   GigabitEthernet3/0/0
          192.168.20.255/32 Direct 0   0        D  127.0.0.1   GigabitEthernet3/0/0
          192.168.21.0/24 Static 60  0        D  192.168.20.2 GigabitEthernet3/0/0
          255.255.255.255/32 Direct 0   0        D  127.0.0.1   InLoopBack0
```

Step 3 Change the priorities of the IPv4 static routes.

Change the priorities of static routes on Device T.

```
<DeviceT> system-view
[~DeviceT] ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.10.2 preference 40
[*DeviceT] commit
[~DeviceT] quit
```

Check the IP routing table of Device T. The following command output shows that the preference value of the IPv4 static route has been changed.

```
<DeviceT> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 16      Routes : 16

Destination/Mask Proto Pre Cost     Flags NextHop       Interface
127.0.0.0/8   Direct 0 0          D 127.0.0.1    InLoopBack0
127.0.0.1/32  Direct 0 0          D 127.0.0.1    InLoopBack0
127.255.255.255/32 Direct 0 0          D 127.0.0.1    InLoopBack0
172.16.1.0/24 Direct 0 0          D 172.16.1.1    GigabitEthernet1/0/0
172.16.1.1/32 Direct 0 0          D 127.0.0.1    GigabitEthernet1/0/0
172.16.1.255/32 Direct 0 0          D 127.0.0.1    GigabitEthernet1/0/0
172.17.1.0/24 Static 40 0          D 192.168.10.2  GigabitEthernet2/0/0
192.168.10.0/24 Direct 0 0          D 192.168.10.1  GigabitEthernet2/0/0
192.168.10.1/32 Direct 0 0          D 127.0.0.1    GigabitEthernet2/0/0
192.168.10.255/32 Direct 0 0          D 127.0.0.1    GigabitEthernet2/0/0
192.168.11.0/24 Static 60 0          D 192.168.10.2  GigabitEthernet2/0/0
192.168.20.0/24 Direct 0 0          D 192.168.20.1  GigabitEthernet3/0/0
192.168.20.1/32 Direct 0 0          D 127.0.0.1    GigabitEthernet3/0/0
192.168.20.255/32 Direct 0 0          D 127.0.0.1    GigabitEthernet3/0/0
192.168.21.0/24 Static 60 0          D 192.168.20.2  GigabitEthernet3/0/0
255.255.255.255/32 Direct 0 0          D 127.0.0.1    InLoopBack0
```

Step 4 Enable FRR for IPv4 static routes.

Enable FRR for static route on Device T.

```
<DeviceT> system-view
[~DeviceT] ip route-static frr
[*DeviceT] commit
[~DeviceT] quit
```

Check the backup outbound interface and the backup next hop on Device T.

```
<DeviceT> display ip routing-table 172.17.1.0 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Summary Count : 1

Destination: 172.17.1.0/24
Protocol: Static      Process ID: 0
Preference: 40          Cost: 0
NextHop: 192.168.10.2 Neighbour: 0.0.0.0
State: Active Adv      Age: 00h00m03s
Tag: 0                  Priority: medium
Label: NULL            QoSInfo: 0x0
IndirectID: 0x31000032
RelayNextHop: 0.0.0.0      Interface: GigabitEthernet2/0/0
TunnelID: 0x0           Flags: D
BkNextHop: 192.168.20.2 BkInterface: GigabitEthernet3/0/0
BkLabel: NULL           SectunnelID: 0x0
BkPETunnelID: 0x0        BkPESecTunnelID: 0x0
BkIndirectID: 0x32000033
```

Step 5 Configure static BFD for IPv4 static routes.

- Configure a BFD session.

On Device T, configure a BFD session between Device T and Device C.

```
<DeviceT> system-view
[~DeviceT] bfd
[*DeviceT-bfd] quit
[*DeviceT] bfd aa bind peer-ip 192.168.11.1 source-ip 192.168.10.1
[*DeviceT-bfd-session-aa] discriminator local 10
[*DeviceT-bfd-session-aa] discriminator remote 20
[*DeviceT-bfd-session-aa] commit
[~DeviceT-bfd-session-aa] quit
```

On Device C, configure a BFD session between Device C and Device T.

```
<DeviceC> system-view
[~DeviceC] bfd
[*DeviceC-bfd] quit
[*DeviceC] bfd ab bind peer-ip 192.168.10.1 source-ip 192.168.11.1
[*DeviceC-bfd-session-ab] discriminator local 20
[*DeviceC-bfd-session-ab] discriminator remote 10
[*DeviceC-bfd-session-ab] commit
[~DeviceC-bfd-session-ab] quit
```

- Configure a static route and bind it to the BFD session.

On Device T, configure a static route and bind it to the BFD session named **aa**.

```
[~DeviceT] ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.10.2 preference 40 track
bfd-session aa
```

Step 6 Simulate a fault on Link A.

```
<DeviceT> system-view
[~DeviceT] interface gigabitethernet 2/0/0
[~DeviceT-GigabitEthernet2/0/0] shutdown
[~DeviceT-GigabitEthernet2/0/0] commit
[~DeviceT-GigabitEthernet2/0/0] quit
[~DeviceT] quit
```

Check the routes to 172.17.1.0/24 on Device T.

```
<DeviceT> display ip routing-table 172.17.1.0 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Summary Count : 1

Destination: 172.17.1.0/24
Protocol: Static      Process ID: 0
Preference: 60          Cost: 0
NextHop: 192.168.20.2   Neighbour: 0.0.0.0
State: Active Adv     Age: 00h00m07s
Tag: 0                 Priority: medium
Label: NULL            QoSInfo: 0x0
IndirectID: 0x32000033
RelayNextHop: 0.0.0.0   Interface: GigabitEthernet3/0/0
TunnelID: 0x0          Flags: D
```

----End

Configuration Files

- Device T configuration file

```
#
sysname DeviceT
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
```

```
undo shutdown
ip address 192.168.10.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.20.1 255.255.255.0
#
bfd aa bind peer-ip 192.168.11.1 source-ip 192.168.10.1
discriminator local 10
discriminator remote 20
#
ip route-static frr
ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.10.2 preference 40 track bfd-session aa
ip route-static 172.17.1.0 24 GigabitEthernet3/0/0 192.168.20.2
ip route-static 192.168.11.0 24 GigabitEthernet2/0/0 192.168.10.2
ip route-static 192.168.21.0 24 GigabitEthernet3/0/0 192.168.20.2
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.10.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.11.2 255.255.255.0
#
ip route-static 172.16.1.0 24 GigabitEthernet1/0/0 192.168.10.1
ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.11.1
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.20.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.21.2 255.255.255.0
#
ip route-static 172.16.1.0 24 GigabitEthernet1/0/0 192.168.20.1
ip route-static 172.17.1.0 24 GigabitEthernet2/0/0 192.168.10.1
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.17.1.0 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.11.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.21.1 255.255.255.0
```

```
#  
bfd ab bind peer-ip 192.168.10.1 source-ip 192.168.11.1  
discriminator local 20  
discriminator remote 10  
#  
ip route-static 172.16.1.0 24 GigabitEthernet2/0/0 192.168.11.2  
ip route-static 172.16.1.0 24 GigabitEthernet3/0/0 192.168.21.2  
ip route-static 192.168.10.0 255.255.255.0 GigabitEthernet2/0/0 192.168.11.2  
ip route-static 192.168.20.0 255.255.255.0 GigabitEthernet3/0/0 192.168.21.2  
#  
return
```

Example for Configuring Association between LDP and Static Routes

On an MPLS network with primary and secondary LSPs established by LSRs using static routes, association between LDP and static routes prevents traffic from being interrupted during traffic switchover or switchback.

Networking Requirements

On an MPLS network with primary and secondary LSPs established by LSRs using static routes, if association between LDP and static routes is not enabled, traffic is interrupted for a short period of time during traffic switchover from the primary LSP to the secondary LSP when the LDP session on the primary link fails (not because of a link fault) or during traffic switchback when the primary link recovers from a fault.

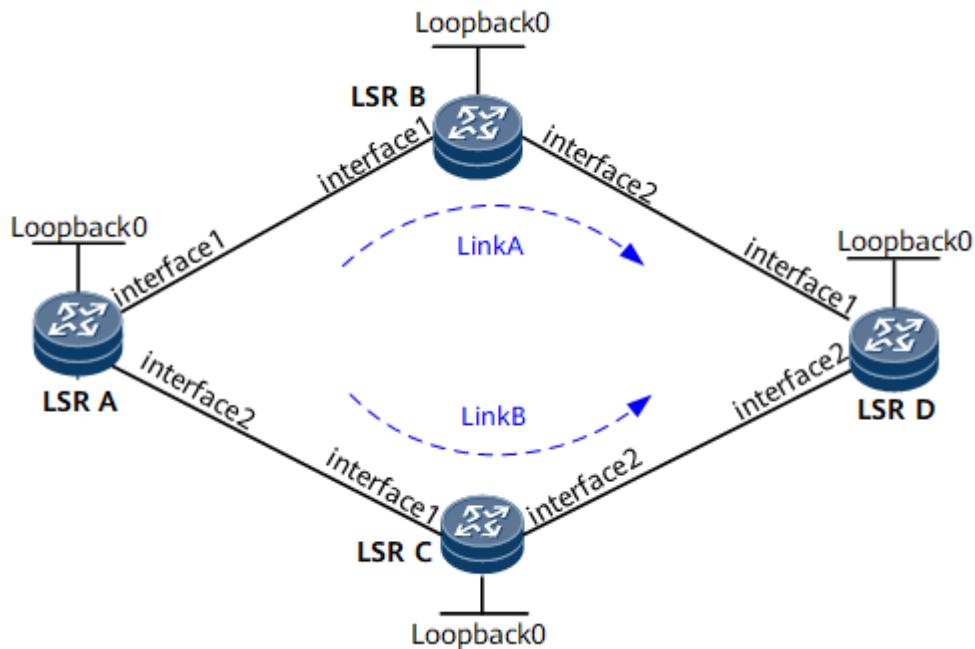
In [Figure 1-50](#), LSRA has two static routes to LSRD. One of the static routes passes through LSRB, whereas the other static route passes through LSRC. LDP LSPs are established based on the static routes. Link A is the primary link, whereas Link B is the backup link. It is required that association between LDP and static routes be configured to prevent MPLS traffic from being interrupted if the LDP session on Link A or Link B recovers from a fault.

Figure 1-50 Networking for configuring association between LDP and static routes



NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Device Name	Interface Name	Interface IP Address	Device Name	Interface Name	Interface IP Address
LSRA	GE 1/0/0	10.1.1.1/30	LSRC	GE 1/0/0	10.2.1.2/30
	GE 2/0/0	10.2.1.1/30		GE 2/0/0	10.4.1.1/30
	Loopback0	1.1.1.1/32		Loopback0	3.3.3.3/32
LSRB	GE 1/0/0	10.1.1.2/30	LSRD	GE 1/0/0	10.3.1.2/30
	GE 2/0/0	10.3.1.1/30		GE 2/0/0	10.4.1.2/30
	Loopback0	2.2.2.2/32		Loopback0	4.4.4.4/32

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure static routes for each LSP to reach other LSRs.
2. Enable MPLS and MPLS LDP globally and on interfaces of LSRs.
3. Configure association between LDP and static routes and check configurations.

Data Preparation

To complete the configuration, you need the following data:

- IP addresses of interfaces on LSRs

- MPLS LSR IDs of LSRs
- Hold-down timer value

Procedure

Step 1 Assign an IP address and mask to each interface.

Configure an IP address for each interface based on [Figure 1-50](#). For configuration details, see "Configuration Files" in this section.

Step 2 Configure static routes on each node to ensure network connectivity.

Configure two static routes with different priorities from LSRA to LSRD, and two static routes with different priorities from LSRD to LSRA.

Configure LSRA.

```
[~LSRA] ip route-static 2.2.2.2 32 GigabitEthernet1/0/0 10.1.1.2
[*LSRA] ip route-static 3.3.3.3 32 GigabitEthernet2/0/0 10.2.1.2
[*LSRA] ip route-static 10.3.1.1 30 GigabitEthernet1/0/0
[*LSRA] ip route-static 10.4.1.1 30 GigabitEthernet2/0/0
[*LSRA] ip route-static 4.4.4.4 32 GigabitEthernet1/0/0 10.1.1.2 preference 40
[*LSRA] ip route-static 4.4.4.4 32 GigabitEthernet2/0/0 10.2.1.2 preference 60
[*LSRA] commit
```

Configure LSRB.

```
[~LSRB] ip route-static 1.1.1.1 32 GigabitEthernet1/0/0 10.1.1.1
[*LSRB] ip route-static 4.4.4.4 32 GigabitEthernet2/0/0 10.3.1.2
[*LSRB] commit
```

Configure LSRC.

```
[~LSRC] ip route-static 1.1.1.1 32 GigabitEthernet1/0/0 10.2.1.1
[*LSRC] ip route-static 4.4.4.4 32 GigabitEthernet2/0/0 10.4.1.2
[*LSRC] commit
```

Configure LSRD.

```
[~LSRD] ip route-static 2.2.2.2 32 GigabitEthernet1/0/0 10.3.1.1
[*LSRD] ip route-static 3.3.3.3 32 GigabitEthernet2/0/0 10.4.1.1
[*LSRD] ip route-static 10.1.1.2 30 GigabitEthernet1/0/0
[*LSRD] ip route-static 10.2.1.2 30 GigabitEthernet2/0/0
[*LSRD] ip route-static 1.1.1.1 32 GigabitEthernet1/0/0 10.3.1.1 preference 40
[*LSRD] ip route-static 1.1.1.1 32 GigabitEthernet2/0/0 10.4.1.1 preference 60
[*LSRD] commit
```

After completing the configurations, run the **display ip routing-table protocol static** command on each node to check the static route configurations. The following example uses the command output on LSRA.

```
[~LSRA] display ip routing-table protocol static
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
public_Routing Table : Static
    Destinations : 5      Routes : 5      Configured Routes : 5

Static routing table status :
    Destinations : 5      Routes : 5

Destination/Mask Proto Pre Cost Flags NextHop      Interface
    2.2.2.2/32 Static 60 0      D 10.1.1.1      GigabitEthernet1/0/0
    3.3.3.3/32 Static 60 0      D 10.2.1.1      GigabitEthernet2/0/0
    4.4.4.4/32 Static 40 0      D 10.1.1.1      GigabitEthernet1/0/0
    10.3.1.0/30 Static 60 0      D 10.1.1.1      GigabitEthernet1/0/0
```

```
10.4.1.0/30 Static 60 0      D 10.2.1.1      GigabitEthernet2/0/0

Static routing table status : <Inactive>
Destinations : 0      Routes : 0
```

Step 3 Enable MPLS LDP on each LSR to set up LDP LSPs.

Configure LSRA.

```
[~LSRA] mpls lsr-id 1.1.1.1
[*LSRA] mpls
[*LSRA-mpls] quit
[*LSRA] mpls ldp
[*LSRA-mpls-ldp] commit
[*LSRA-mpls-ldp] quit
[~LSRA] interface GigabitEthernet1/0/0
[~LSRA-GigabitEthernet1/0/0] mpls
[*LSRA-GigabitEthernet1/0/0] mpls ldp
[*LSRA-GigabitEthernet1/0/0] commit
[*LSRA-GigabitEthernet1/0/0] quit
[~LSRA] interface GigabitEthernet2/0/0
[~LSRA-GigabitEthernet2/0/0] mpls
[*LSRA-GigabitEthernet2/0/0] mpls ldp
[*LSRA-GigabitEthernet2/0/0] commit
[*LSRA-GigabitEthernet2/0/0] quit
```

Repeat this step for LSRB, LSRC, and LSRD. For configuration details, see "Configuration Files" in this section.

Run the **display mpls ldp session** command on each node to view information about established LDP sessions (in **Operational** state). The following example uses the command output on LSRA.

```
[~LSRA] display mpls ldp session
LDP Session(s) in Public Network
Codes: LAM(Label Advertisement Mode), SsnAge Unit(DDDD:HH:MM)
An asterisk (*) before a session means the session is being deleted.

PeerID      Status    LAM  SsnRole  SsnAge      KASent/Rcv
-----
2.2.2.2:0   Operational DU  Passive  0000:15:34  3738/3738
3.3.3.3:0   Operational DU  Passive  0000:00:45  182/182
-----
TOTAL: 2 Session(s) Found.
```

Step 4 Configure association between LDP and static routes on LSRA and LSRD.

Configure LSRA.

```
[~LSRA] ip route-static 4.4.4.4 32 GigabitEthernet1/0/0 10.1.1.2 preference 40 ldp-sync
[*LSRA] interface GigabitEthernet1/0/0
[*LSRA-GigabitEthernet1/0/0] static-route timer ldp-sync hold-down 20
[*LSRA-GigabitEthernet1/0/0] commit
[~LSRA-GigabitEthernet1/0/0] quit
```

Configure LSRD.

```
[~LSRD] ip route-static 1.1.1.1 32 GigabitEthernet1/0/0 10.3.1.1 preference 40 ldp-sync
[*LSRD] interface GigabitEthernet1/0/0
[*LSRD-GigabitEthernet1/0/0] static-route timer ldp-sync hold-down 20
[*LSRD-GigabitEthernet1/0/0] commit
[~LSRD-GigabitEthernet1/0/0] quit
```

Step 5 Verify the configuration.

Display outbound interface status of the static route that is associated with LDP on LSRA.

```
[~LSRA] display static-route ldp-sync
```

```
Total number of routes enable Ldp-Sync: 1
-----
Interface GigabitEthernet1/0/0
Enable ldp-sync static routes number: 1
Static-route ldp-sync holddown timer: 20s
Sync state: Normal
Dest = 4.4.4.4, Mask = 32, NextHop = 10.1.1.2.
```

The command output shows that association between LDP and static routes has been configured (in **Normal** state).

- If the LDP session on Link A fails, traffic is switched to Link B immediately to ensure association between LDP and static routes and prevent a traffic interruption.
- If Link A fails and then recovers, the static route with the next hop of 10.1.1.2 is not preferentially selected until the hold-down timer (20 seconds) expires (by then the LDP session on Link A has been set up). Then traffic is switched back to Link A. In this way, association between LDP and static routes is ensured, and MPLS traffic is not interrupted.

----End

Configuration Files

- LSRA configuration file

```
#  
sysname LSRA  
#  
mpls lsr-id 1.1.1.1  
#  
mpls  
#  
mpls ldp  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.1.1 255.255.255.252  
static-route timer ldp-sync hold-down 20  
mpls  
mpls ldp  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.2.1.1 255.255.255.252  
mpls  
mpls ldp  
#  
interface LoopBack0  
ip address 1.1.1.1 255.255.255.255  
#  
ip route-static 2.2.2.2 255.255.255.255 GigabitEthernet1/0/0 10.1.1.2  
ip route-static 3.3.3.3 255.255.255.255 GigabitEthernet2/0/0 10.2.1.2  
ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet1/0/0 10.1.1.2 preference 40 ldp-sync  
ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet2/0/0 10.2.1.2  
ip route-static 10.3.1.0 255.255.255.252 GigabitEthernet1/0/0  
ip route-static 10.4.1.0 255.255.255.252 GigabitEthernet2/0/0  
#  
return
```

- LSRB configuration file

```
#  
sysname LSRB  
#  
mpls lsr-id 2.2.2.2  
#
```

```
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.252
mpls
mpls ldp
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.3.1.1 255.255.255.252
mpls
mpls ldp
#
interface LoopBack0
ip address 2.2.2.2 255.255.255.255
#
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/0 10.1.1.1
ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet2/0/0 10.3.1.2
#
return
```

- LSRC configuration file

```
#
sysname LSRC
#
mpls lsr-id 3.3.3.3
#
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.252
mpls
mpls ldp
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.4.1.1 255.255.255.252
mpls
mpls ldp
#
interface LoopBack0
ip address 3.3.3.3 255.255.255.255
#
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/0 10.2.1.1
ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet2/0/0 10.4.1.2
#
return
```

- LSRD configuration file

```
#
sysname LSRD
#
mpls lsr-id 4.4.4.4
#
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.3.1.2 255.255.255.252
static-route timer ldp-sync hold-down 20
mpls
mpls ldp
```

```
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.4.1.2 255.255.255.252  
mpls  
mpls ldp  
#  
interface LoopBack0  
ip address 4.4.4.4 255.255.255.255  
#  
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/0 10.3.1.1 preference 40 ldp-sync  
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet2/0/0 10.4.1.1  
ip route-static 2.2.2.2 255.255.255.255 GigabitEthernet1/0/0 10.3.1.1  
ip route-static 3.3.3.3 255.255.255.255 GigabitEthernet2/0/0 10.4.1.1  
ip route-static 10.1.1.0 255.255.255.252 GigabitEthernet1/0/0  
ip route-static 10.2.1.0 255.255.255.252 GigabitEthernet2/0/0  
#  
return
```

1.1.3 IPv6 Static Route Configuration

1.1.3.1 IPv6 Static Route Configuration

Static routes are applicable to networks with simple structures. Properly configuring and using static routes can improve network performance and guarantee the required bandwidth for important applications.

1.1.3.1.1 Overview of IPv6 Static Routes

Configuring IPv6 static routes can implement the interworking of simple networks.

Definition

Static routes are special routes that are configured by network administrators.

Purpose

On a simple network, only static routes can ensure that the network runs properly. If a router cannot run dynamic routing protocols or cannot generate routes to a destination network, you can configure static routes on the router.

Route selection can be controlled using static routes. Properly configuring and using static routes can improve network performance and guarantee the required bandwidth for important applications. When a network fault occurs or the network topology changes, however, static routes must be changed manually by the administrator.

1.1.3.1.2 Configuration Precautions for IPv6 Static Route

Feature Requirements

None

1.1.3.1.3 Configuring IPv6 Static Routes

On a network, you can control route selection by configuring IPv6 static routes.

Usage Scenario

On a small IPv6 network, you can achieve network connectivity by configuring IPv6 static routes. Compared with the use of dynamic routing protocols, configuring static routes saves bandwidth resources.

Pre-configuration Tasks

Before configuring an IPv6 static route, configure link layer protocol parameters and IPv6 addresses for interfaces and ensure that the status of the link layer protocol of the interface is Up.

Creating IPv6 Static Routes

To create an IPv6 static route, configure its destination IP address, outbound interface, and next hop.

Context

When creating a static route, you can specify an outbound interface, a next hop address, or both of them, depending on actual requirements.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Configure an IPv6 static route.

- Run one of the following commands to configure an IPv6 static route for the public network:
 - **ipv6 route-static** *dest-ipv6-address prefix-length interface-type interface-number* [*nexthop-ipv6-address*] [**preference** *preference* | **tag** *tag*] [**description** *text*]
 - **ipv6 route-static** *dest-ipv6-address prefix-length nexthop-ipv6-address* [**preference** *preference* | **tag** *tag*] * [**description** *text*]
 - **ipv6 route-static** *dest-ipv6-address prefix-length vpn-instance vpn-instance-name nexthop-ipv6-address* [**preference** *preference* | **tag** *tag*] * [**description** *text*]
 - **ipv6 route-static** *dest-ipv6-address prefix-length vpn-instance vpn-instance-name* [**preference** *preference* | **tag** *tag*] * [**description** *text*]
- Run one of the following commands to configure an IPv6 static route for a VPN instance:
 - **ipv6 route-static vpn-instance** *vpn-source-name dest-ipv6-address prefix-length* { *interface-name* | *interface-type interface-number* } [*nexthop-ipv6-address*] [**preference** *preference* | **tag** *tag*] * [**description** *text*]
 - **ipv6 route-static vpn-instance** *vpn-source-name dest-ipv6-address prefix-length nexthop-ipv6-address* [**preference** *preference* | **tag** *tag*] * [**description** *text*]

- **ipv6 route-static vpn-instance vpn-source-name dest-ipv6-address prefix-length { vpn-instance vpn-instance-name nexthop-ipv6-address | nexthop-ipv6-address [public] } [preference preference | tag tag] * [description text]**
- **ipv6 route-static vpn-instance vpn-source-name dest-ipv6-address prefix-length { vpn-instance vpn-instance-name | public } [preference preference | tag tag] * [description text]**
- To configure an IPv6 static route in the topology instance, run the **ipv6 route-static topology topology-name dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | nexthop-ipv6-address } [preference preference | tag tag] * [no-advertise | no-install] [description text]** command.

Step 3 Run commit

The configuration is committed.

----End

(Optional) Setting the Default Priority for IPv6 Static Routes

You can change default priority for IPv6 static routes.

Context

After an IPv6 static route is configured, the default priority is used if no priority is specified for the static route. After the default priority is re-set, the new default priority takes effect only on new IPv6 static routes.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **ipv6 route-static default-preference preference**

The default preference value is configured for IPv6 static routes.

By default, the priority of IPv6 static routes is lower than those of OSPF and IS-IS routes. To allow IPv6 static routes to take effect when OSPF or IS-IS routes exist, configure the default priority of IPv6 static routes to be higher than that of OSPF or IS-IS routes before you configure IPv6 static routes. A smaller preference value indicates a higher priority.

Step 3 Run commit

The configuration is committed.

----End

(Optional) Enabling a Device to Recurse IPv6 Static Routes to ND Vlink Direct Routes

To prevent blackhole routes from being generated in a scenario where a Layer 2 network accesses a Layer 3 network, you can enable a device to recurse IPv6 static routes to ND Vlink direct routes.

Context

In a scenario where a Layer 2 network accesses a Layer 3 network, you can enable a device to recurse IPv6 static routes to ND Vlink direct routes. This prevents blackhole routes from being generated.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ipv6 route recursive-lookup nd vlink-direct-route protocol static**

The device is configured to recurse IPv6 static routes to ND Vlink direct routes.

The **ipv6 route recursive-lookup nd vlink-direct-route protocol static** command is mainly used in scenarios where Layer 2 accesses Layer 3. To allow IPv6 static routes to recurse only to ND Vlink direct routes, run the **ipv6 route-static** command with the **recursive-lookup host-route** parameter specified in addition to running the **ipv6 route recursive-lookup nd vlink-direct-route protocol static** command.

Step 3 Run **commit**

The configuration is committed.

----End

(Optional) Enabling a Device to Recurse IPv6 Static Routes to SRv6 Routes

By default, a device does not recurse static routes to SRv6 routes. In EVPN L3VPN HoVPN over SRv6 BE or EVPN L3VPN HoVPN over SRv6 TE Policy scenarios, you can enable a device to recurse static routes to SRv6 routes to prevent traffic black holes.

Context

In an EVPN L3VPN HoVPN over SRv6 BE or EVPN L3VPN HoVPN over SRv6 TE Policy scenario, if the next hop of a default route on an SPE is the SPE itself and the link between the SPE and the corresponding NPE fails, the UPE cannot detect the link fault. As a result, a traffic black hole occurs after traffic reaches the SPE. To resolve this problem, you need to specify the NPE's address (public or private) as the next-hop address of the default static route on the SPE. In this manner, the validity of the static route depends on whether the link between the SPE and NPE is available. In addition, you need to enable the SPE to recurse static routes to SRv6 routes. This configuration allows the SPE to withdraw this inactive route when the SPE-NPE link fails so that the UPE can detect the route withdrawal and no longer sends traffic to the SPE, thereby preventing a traffic black hole.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run `ipv6 route-static recursive-lookup inherit-label-route segment-routing-ipv6`

The device is enabled to recurse IPv6 static routes to SRv6 routes.

Step 3 Run `commit`

The configuration is committed.

----End

(Optional) Preventing an IPv6 Static Route from Being Selected If the BFD Session Associated with It Is in the AdminDown State

This section describes how to configure the router not to select an IPv6 static route if the BFD session associated with it is in the AdminDown state. This ensures that Huawei devices can interwork with non-Huawei devices.

Context

By default, an IPv6 static route can still be selected by Huawei devices even though the BFD session associated with it is in the AdminDown state, but not by non-Huawei devices. As a result, Huawei devices cannot interwork with non-Huawei devices.

To address this problem, run the **`ipv6 route-static track bfd-session admindown invalid`** command to configure the router not to select the IPv6 static route if the BFD session associated with it is in the AdminDown state.

Procedure

Step 1 Run `system-view`

The system view is displayed.

Step 2 Run `ipv6 route-static track bfd-session session-name bfd-name admindown invalid`

The router has been configured not to select the IPv6 static route if the BFD session associated with it is in the AdminDown state.

Step 3 Run `commit`

The configuration is committed.

----End

Verifying the IPv6 Static Route Configuration

After configuring an IPv6 static route, verify the configuration.

Prerequisites

An IPv6 static route has been configured.

Procedure

- Run the **display ipv6 routing-table** command to check brief information about the IPv6 routing table.
- Run the **display ipv6 routing-table verbose** command to check detailed information about the IPv6 routing table.
- Run the **display ipv6 routing-table protocol** command to check information about routes of a specified routing protocol.

----End

1.1.3.1.4 Configuring IPv6 Floating Static Routes

Configuring floating static routes helps to implement route backup and improve network reliability.

Usage Scenario

If there is a route to a specific destination, you can configure a static route with a low preference to implement route backup, which improves network reliability. This static route with a lower preference is called a floating static route. It is activated to forward packets only when the optimal route fails. When the optimal route recovers, this static route becomes inactive.

Floating static routes are used as follows:

- Two static routes with the same destination address but different priorities are configured, and the one with the lower priority functions as a backup.
- When a route of a dynamic routing protocol to a destination address exists, you can configure a static route to the destination address for data service forwarding temporarily if the dynamic routing protocol restarts.

Pre-configuration Tasks

Before configuring IPv6 floating static routes, configure link layer protocol parameters and IP addresses for interfaces and ensure that the status of the link layer protocol on the interfaces is Up.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run either of the following commands as required:

- Run **ipv6 route-static dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name [nexthop-ipv6-address] | nexthop-ipv6-address } [preference preference | tag tag]* [description text]**
An IPv6 floating static route is configured in public network.
- Run **ipv6 route-static vpn-instance vpn-source-name dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name [nexthop-ipv6-address] | nexthop-ipv6-address [public] } [preference preference | tag tag]* [description text]**

An IPv6 floating static route is configured in VPN network.

- Run **ipv6 route-static topology topology-name dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | nexthop-ipv6-address } [preference preference | tag tag] * [description text]**

An IPv6 floating static route is configured in a non-base topology instance.

The **preference** parameter specifies a route priority. The greater the value, the lower the priority. Therefore, make sure that the priority value of the floating static route is greater than that of the optimal route.

Step 3 Run commit

The configuration is committed.

----End

Checking the Configurations

After the configuration is complete, check the configuration results.

- Run the **display ipv6 routing-table ipv6-address [prefix-length]** command to check configurations about the optimal IPv6 static route to the specified destination network segment.
- Run the **display current-configuration | include static** command to check the configurations about the current static route.

1.1.3.1.5 Configuring IPv6 Static Route Detection

With IPv6 static route detection, if a link through which an IPv6 static route travels fails, a rapid link switchover is performed, preventing lengthy service interruptions.

Usage Scenario

Unlike dynamic routes, static routes do not have a detection mechanism. As a result, administrator intervention is required if a fault occurs on the network. To prevent the need for administrator intervention, deploy IPv6 static route detection to implement automatic rapid link switchover if a link failure occurs. IPv6 static route detection prevents lengthy service interruptions.

The following IPv6 static route detection methods are available:

- BFD for IPv6 static routes: After BFD for IPv6 static routes is configured, each static route can be bound to a BFD session. BFD for IPv6 static routes implements millisecond-level detection. BFD is classified as static BFD or dynamic BFD.
- Network quality analysis (NQA) for IPv6 static routes: Although BFD for IPv6 static routes implements fast link fault detection, it cannot be deployed in some scenarios (when Layer 2 devices exist on the network, for example) because both ends of the link must support BFD. NQA for IPv6 static routes, however, can monitor the link status of a static route even when only one end supports NQA. NQA for IPv6 static routes implements second-level detection.

NOTE

Only one of the IPv6 static route detection methods can be deployed. Therefore, choose one based on the live network requirements.

Pre-configuration Tasks

Before configuring IPv6 static route detection, configure link-layer protocol parameters and IPv6 addresses for interfaces to ensure that the link layer protocol of each interface is Up.

Configuring Dynamic BFD for IPv6 Static Routes

Dynamic BFD for IPv6 static routes enables devices to fast detect link changes, improving network reliability.

Usage Scenario

To use BFD sessions to provide link detection for IPv6 static routes on the public network, you can bind static routes to BFD sessions. One IPv6 static route can be bound to one BFD session.

Optimal IPv6 static routes are delivered to the forwarding table for packet forwarding. However, IPv6 static routes cannot detect the status of the link to the next hop. You can bind IPv6 static routes to BFD sessions. A BFD session can fast detect changes over a link and inform the routing management system of the changes. The routing management system immediately deletes the IPv6 static route that is bound to the BFD session from the forwarding table and recalculates another active route. In this manner, fast route convergence is implemented.

Pre-configuration Tasks

Before configuring dynamic BFD for IPv6 static routes, configure parameters of the link layer protocol and IPv6 addresses for interfaces and ensure that the link layer protocol on the interfaces is up.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bfd**

BFD is enabled globally on the local device.

NOTE

Running the **undo bfd** command will delete the parameters of the BFD session bound to the static route. As a result, the static route status may change, and services may be interrupted.

Step 3 Run **quit**

Return to the system view.

Step 4 (Optional) Run **ipv6 route-static default-bfd { detect-multiplier multiplier | min-rx-interval min-rx-interval | min-tx-interval min-tx-interval }***

Global BFD parameters are configured for IPv6 static routes.

Step 5 Run **ipv6 route-static bfd [interface-type interface-number | vpn-instance vpn-destination-name] nexthop-address [local-address ipv6-address] [detect-**

multiplier multiplier] [min-rx-interval min-rx-interval] [min-tx-interval min-tx-interval]*

BFD parameters are set for a single IPv6 static route.

 NOTE

If the *interface-type interface-number* parameter is not set, the **local-address ipv6-address** parameter must be specified.

If none of **min-rx-interval**, **min-tx-interval**, or **detect-multiplier** is specified, the default values of the global BFD parameters are used.

Step 6 Run either of the following commands as required:

- Run **ipv6 route-static dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name [nexthop-ipv6-address] | nexthop-ipv6-address } [preference preference | tag tag]* bfd enable [description text]**
A BFD session is bound to an IPv6 static route on the public network.
- Run **ipv6 route-static vpn-instance vpn-source-name dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name [nexthop-ipv6-address] | nexthop-ipv6-address [public] } [preference preference | tag tag]* bfd enable [description text]**
A BFD session is bound to an IPv6 static route on the VPN network.

 NOTE

The outbound interface and next hop IP address specified when binding the IPv6 static route to a BFD session must be the same as those specified when configuring BFD parameters for the IPv6 static route.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the following commands to check the previous configuration.

- Run the **display bfd session { all | discriminator descr-value } [verbose]** command to check information about BFD sessions.
- Run the **display current-configuration | include bfd** command to check configurations of BFD for IPv6 static routes.

Information about a BFD session can be viewed only after parameters of the BFD session are set and the BFD session is established.

If a BFD session has been established, the status of the BFD session is Up. Run the **display current-configuration | include bfd** command in the system view, and you can see that the BFD session has been bound to a static route.

Configuring Static BFD for IPv6 Static Routes

Static BFD for IPv6 static routes enables devices to fast detect link changes, improving network reliability.

Usage Scenario

To use BFD sessions to provide link detection for IPv6 static routes on the public network, you can bind IPv6 static routes to BFD sessions. One IPv6 static route can be bound to one BFD session.

Optimal IPv6 static routes are delivered to the forwarding table for packet forwarding. However, IPv6 static routes cannot detect the status of the link to the next hop. You can bind IPv6 static routes to BFD sessions. A BFD session can fast detect changes over a link and inform the routing management system of the changes. The routing management system immediately deletes the static route that is bound to the BFD session from the forwarding table and recalculates another active route. In this manner, fast route convergence is implemented.

Pre-configuration Tasks

Before configuring static BFD for IPv6 static routes, complete the following tasks:

- Configure link layer protocol parameters and IP addresses for interfaces and ensure that the link layer protocol on the interfaces is up.

Procedure

Step 1 Configure a BFD Session.

- Run **system-view**
The system view is displayed.
- Run **bfd**
BFD is globally enabled.
- Run **quit**
The system view is displayed.
- Run **bfd session-name bind peer-ipv6 peer-ipv6**
The binding between a BFD session for IPv6 and a peer IPv6 address is created, and the BFD session view is displayed.
- Run **discriminator local discr-value**
The local discriminator of a static BFD session is set.
- Run **discriminator remote discr-value**
The remote discriminator of a static BFD session is set.

NOTE

For details of optional procedures for configuring a BFD session, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - Network Reliability*.

- Run **commit**
The configuration is committed.

Step 2 Associate an IPv6 static route with a BFD session.

- Run **system-view**
The system view is displayed.

2. Run either of the following commands as required:
 - Run **ipv6 route-static dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-instance-name nexthop-ipv6-address | nexthop-ipv6-address } [preference preference | tag tag] * track bfd-session cfg-name [description text]**
A BFD session is bound to an IPv6 static route on the public network.
 - Run **ipv6 route-static vpn-instance vpn-source-name dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name [nexthop-ipv6-address] | nexthop-ipv6-address [public] } [preference preference | tag tag] * track bfd-session cfg-name [description text]**
A static BFD session is bound to an IPv6 static route on the VPN network.
3. Run **commit**
The configuration is committed.
----End

Verifying the Configuration

After configuring static BFD for IPv6 static routes, verify the configuration.

- Run the **display bfd session { all | discriminator discr-value } [verbose]** command to check information about BFD sessions.
- Run the **display current-configuration | include bfd** command to check configurations of BFD for IPv6 static routes.

Information about a BFD session can be viewed only after parameters of the BFD session are set and the BFD session is established.

If a BFD session has been established, the status of the BFD session is up. Run the **display current-configuration | include bfd** command in the system view, and you can see that the BFD session has been bound to a static route.

Configuring NQA for IPv6 Static Routes

If an IPv6 static route is associated with a network quality analysis (NQA) test instance, NQA tests the link status periodically. If NQA detects a fault along the associated IPv6 static route, the IPv6 static route is deleted, and traffic is switched to another route.

Usage Scenario

On live networks, the link status of IPv6 static routes must be monitored in real time so that a link switchover can be performed immediately if a link fails. Bidirectional Forwarding Detection (BFD) for IPv6 static routes can implement detection within milliseconds. However, BFD for IPv6 static routes requires that both ends of a link support BFD.

NQA for IPv6 static routes can monitor the link status of static routes as long as one end supports NQA.

If a link fails, an NQA test instance immediately detects the fault and instructs the routing management module to delete the IPv6 static route associated with the

NQA test instance from the IPv6 routing table. Traffic is then forwarded over another link.

 NOTE

Currently, NQA test instances of only the ICMP or TCP type can be associated with static routes to implement fast fault detection.

Pre-configuration Tasks

Before configuring NQA for IPv6 static routes, configure parameters of the link layer protocol and IPv6 addresses for interfaces and ensure that the link layer protocol on the interfaces is up.

Procedure

Step 1 Configure an NQA test instance of the ICMP or TCP type.

1. Run **system-view**

The system view is displayed.

2. Run **nqa test-instance admin-name test-name**

An NQA test instance is created, and the test instance view is displayed.

 NOTE

Running the **undo nqa all-test-instance** command will delete the parameters of the NQA test instance bound to the static route. As a result, the static route status may change, and services may be interrupted.

3. Run **test-type { icmp | tcp }**

The type of the test instance is set to ICMP or TCP.

4. Run **destination-address ipv6 destAddress6**

A destination address is configured.

In an NQA test instance, you can specify an NQA server by running the **destination-address** command to configure a destination address for the NQA test instance.

5. Run any of the following commands to start an NQA test instance:

- **start at [yyyy/mm/dd] hh:mm:ss [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start delay { seconds second | hh:mm:ss } [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start now [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start daily hh:mm:ss to hh:mm:ss [begin { yyyy/mm/dd | yyyy-mm-dd }] [end { yyyy/mm/dd | yyyy-mm-dd }]**

 NOTE

For details of optional procedures when configuring an ICMP NQA test instance, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - System Monitor*.

6. Run **commit**

The configuration is committed.

Step 2 Associate an IPv6 static route with an NQA test instance.

1. Run **system-view**

The system view is displayed.

2. Run either of the following commands as required to associate an IPv6 static route with an NQA test instance:

- To associate an IPv6 static route with an NQA test instance on the public network, run the **ipv6 route-static dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name nexthop-ipv6-address | vpn-instance vpn-destination-name | nexthop-ipv6-address } [preference preference | tag tag] * track nqa admin-name test-name [description text]** command.
- To associate an IPv6 static route with an NQA test instance in a VPN instance, run the **ipv6 route-static vpn-instance vpn-source-name dest-ipv6-address prefix-length { interface-type interface-number [nexthop-ipv6-address] | vpn-instance vpn-destination-name nexthop-ipv6-address | vpn-instance vpn-destination-name | public | nexthop-ipv6-address [public] } [preference preference | tag tag] * track nqa admin-name test-name [description text]** command.

3. Run **commit**

The configuration is committed.

NOTE

Precautions for associating a static route with an NQA test instance are as follows:

- The destination address of an NQA test instance cannot be the destination address of an associated static route.
- If the static route associated with one NQA test instance is associated with another NQA test instance, the association between the static route and the original NQA test instance is deleted.
- An NQA test instance must have been created before you associate it with a static route.

----End

Verifying the Configuration

After configuring NQA for IPv6 static routes, verify the configuration.

- Run the **display current-configuration | include nqa** command to check the configurations of NQA for IPv6 static routes.
- Run the **display nqa results [test-instance admin-name test-name]** command to check the NQA test results.

NOTE

The NQA test results cannot be displayed automatically. You need to run the **display nqa results** command to view the results.

Configuring NQA Group for IPv6 Static Route

After an IPv6 static route is associated with an NQA group, if the NQA group detects a link fault, it withdraws the IPv6 static route, triggering traffic switching and ensuring network stability.

Usage Scenario

On live networks, the status of IPv6 static routes needs to be monitored in real time to ensure network stability. If the link status changes, a primary/backup link switchover is performed. NQA for IPv6 static route can monitor only a single link. To enable a static route to use NQA test instances to monitor multiple links, you can bind these NQA test instances to the same NQA group and associate the static route with the NQA group.

If a link fault occurs, the NQA group can collect statistics on the status changes of all NQA test instances bound to the NQA group and determine whether to change its status according to the configured operation. When the status of the NQA group changes, the NQA group instructs the RM module to delete the IPv6 static route associated with the NQA group from the IP routing table. Traffic is then forwarded along another path, implementing a link switchover.

Pre-configuration Tasks

Before configuring NQA group for IPv6 static route, complete the following task:

- Configure link-layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol of each interface is up.

Procedure

- Configure multiple NQA test instances of the ICMP or TCP type.
 - Run **system-view**
The system view is displayed.
 - Run **nqa test-instance admin-name test-name**
An NQA test instance is created, and the test instance view is displayed.
 - Run **test-type { icmp | tcp }**
The test instance type is set to ICMP or TCP.
 - Run **destination-address ipv6 destAddress6**
A destination address is configured.
In an NQA test instance, you can specify the destination end by running the **destination-address** command to configure a destination address for the NQA test instance.
 - Run any of the following commands to start an NQA test instance:
 - start at [yyyy/mm/dd] hh:mm:ss [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
 - start delay { seconds second | hh:mm:ss } [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**

- **start now [end { at [yyyy/mm/dd] hh:mm:ss | delay { seconds second | hh:mm:ss } | lifetime { seconds second | hh:mm:ss } }]**
- **start daily hh:mm:ss to hh:mm:ss [begin { yyyy/mm/dd | yyyy-mm-dd }] [end { yyyy/mm/dd | yyyy-mm-dd }]**

 NOTE

For more optional configurations of NQA test instances, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - System Monitor*.

- f. Run **commit**
The configuration is committed.
 - g. Repeat the preceding steps to configure multiple NQA test instances of the ICMP or TCP type.
- Bind the preceding NQA test instances to an NQA group.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **nqa group group-name**
An NQA group is created, and the NQA group view is displayed.
 - c. Run **nqa test-instance admin-name test-name**
An NQA test instance is bound to the NQA group.
Run this command multiple times to bind multiple NQA test instances to the NQA group.

 NOTE

The NQA test instances bound to the same NQA group must be of the same type: ICMP or TCP.

- d. Run **operator { and | or }**
The operation type between test instances in the NQA group is set to AND or OR.
By default, the operation type between test instances is **or**.
 - e. (Optional) Run **description string**
A description is configured for the NQA group.
 - f. Run **commit**
The configuration is committed.
- Bind an IPv6 static route to the NQA group.
 - a. Run **system-view**
The system view is displayed.
 - b. Bind an IPv6 static route to the NQA group.
 - To configure an IPv6 static route on the public network and associate the route with the NQA group, run the **ipv6 route-static ip-address { mask | mask-length } { nexthop-address | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name**

*nexthop-address } [preference preference | tag tag] * track nqa-group group-name [description text] command.*

- To configure an IPv6 static route on a VPN and associate the route with the NQA group, run the **ipv6 route-static vpn-instance vpn-source-name destination-address { mask | mask-length } { nexthop-address [public] | interface-type interface-number [nexthop-address] | vpn-instance vpn-instance-name nexthop-address } [preference preference | tag tag] * track nqa-group group-name [description text] command.**

c. Run **commit**

The configuration is committed.

 **NOTE**

When configuring NQA group for IPv6 static route, pay attention to the following points:

- The destination address of an NQA group cannot be the destination address of an associated static route.
- If a static route is associated with another NQA group, the association between the static route and the current NQA group is removed.
- A static route cannot be associated with an NQA group that has not been created.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display static-route ipv6 routing-table** command to check information about IPv6 static routes.
- Run the **display current-configuration | include nqa-group** command to check the configurations of NQA group for IPv6 static route.
- Run the **display nqa group [group-name]** command to check the test result of the NQA group.

 **NOTE**

The test result of the NQA group cannot be displayed automatically. You need to run the **display nqa group** command to view the result.

1.1.3.1.6 Configuring FRR for IPv6 Static Routes

Fast ReRoute (FRR) is applicable to services that are sensitive to packet delay and packet loss. FRR can be configured for IPv6 static routes to protect traffic using backup links.

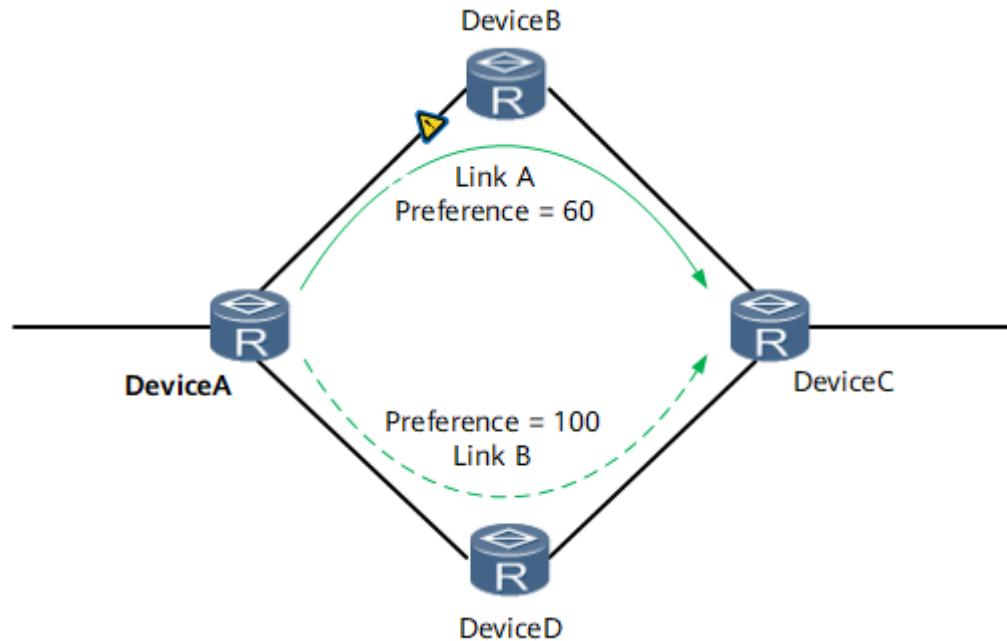
Usage Scenario

FRR is applicable to IP services that are sensitive to delay and packet loss. FRR minimizes the impact of link faults on services.

If two static routes with the same prefix but different priorities are configured, they can implement FRR. The route with the higher priority is the primary route,

and the route with the lower priority is the backup route. If the link that the primary route passes fails, traffic is rapidly switched to the backup route, thereby reducing the number of lost packets. On the network shown in [Figure 1-51](#), two static routes with different priorities from DeviceA to DeviceC work in primary/backup mode. The two routes are both delivered to the forwarding table. When the primary link A is normal, traffic is transmitted through link A. If link A fails, FRR allows traffic to be quickly switched to link B.

[Figure 1-51](#) FRR for IPv6 static routes



Pre-configuration Tasks

Before configuring FRR for IPv6 static routes, complete the following tasks:

- Configure parameters of the link layer protocol and IPv6 addresses for interfaces and ensure that the link layer protocol on the interfaces is up.
- [Configure dynamic BFD for IPv6 static routes](#) or [configure static BFD for IPv6 static routes](#) to speed up fault detection.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **ipv6 route-static frr [vpn-instance vpn-instance-name]**

FRR is enabled for public or VPN IPv6 static routes.

NOTE

To implement route backup by configuring FRR for static routes, you need to specify different priorities for these static routes.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

After configuring FRR for IPv6 static routes, verify the configuration.

- Run the **display ipv6 routing-table verbose** command to check detailed information about the backup outbound interface and the backup next hop in the routing table.
- Run the **display ipv6 routing-table *ipv6-address* [prefix-length] [longer-match] verbose** command to check detailed information about the backup outbound interface and the backup next hop in the routing table.

1.1.3.1.7 Configuration Examples for IPv6 Static Routes

This section provides configuration examples of IPv6 static routes.

Example for Configuring IPv6 Static Routes

You can configure IPv6 static routes to interconnect any two devices on an IPv6 network.

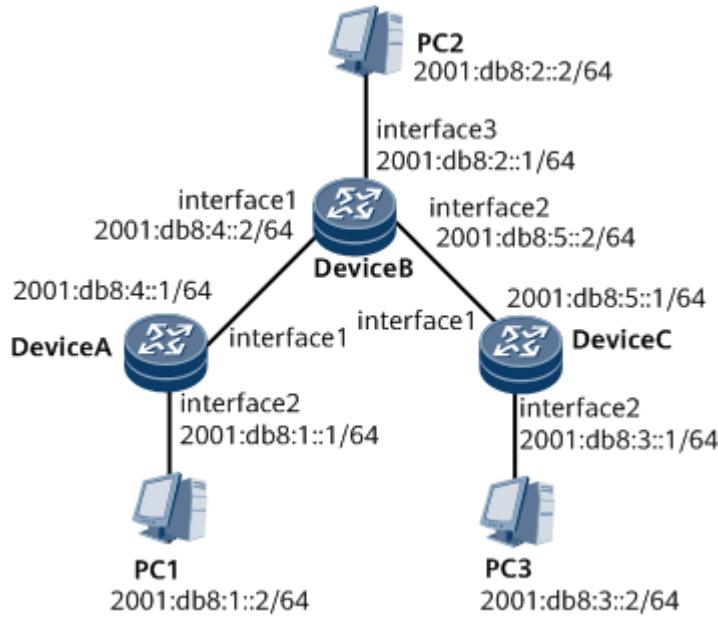
Networking Requirements

On the network shown in [Figure 1-52](#), the prefix of all the IPv6 addresses is 64. It is required that IPv6 static routes be configured between routers to ensure that all hosts communicate with routers. GE interfaces on routers use IPv6 link-local addresses.

Figure 1-52 Networking for configuring IPv6 static routes



Interfaces 1 through 3 in this example represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



Precautions

When configuring an IPv6 static route, specify a next hop address if the outbound interface is of the broadcast type.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IPv6 addresses for GE interfaces on each router.
2. Configure the IPv6 static route and the default route on each router.
3. Configure the IPv6 default gateway on each host so that any two hosts can communicate with each other.

Data Preparation

To complete the configuration, you need the following data:

- Default route with GE 1/0/0 as the outbound interface on Device A
- Static route to 2001:db8:1:: 64 with GE 1/0/0 as the outbound interface on Device B
- Static route to 2001:db8:3:: 64 with GE 2/0/0 as the outbound interface on Device B
- Default route with GE 1/0/0 as the outbound interface on Device C
- Default gateways of PC1, PC2, and PC3

Procedure

Step 1 Configure an IPv6 address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure IPv6 static routes.

Configure an IPv6 default route on Device A.

```
[~DeviceA] ipv6 route-static :: 0 gigabitethernet 1/0/0 2001:db8:4::2
[*DeviceA] commit
```

Configure two IPv6 static routes on Device B.

```
[~DeviceB] ipv6 route-static 2001:db8:1:: 64 gigabitethernet 1/0/0 2001:db8:4::1
[*DeviceB] ipv6 route-static 2001:db8:3:: 64 gigabitethernet 2/0/0 2001:db8:5::1
[*DeviceB] commit
```

Configure an IPv6 default route on Device C.

```
[~DeviceC] ipv6 route-static :: 0 gigabitethernet 1/0/0 2001:db8:5::2
[*DeviceC] commit
```

Step 3 Configure host addresses and gateways.

Configure IPv6 addresses for hosts, and then configure the default gateways of PC1, PC2, and PC3 as 2001:db8:1::1, 2001:db8:2::1, and 2001:db8:3::1, respectively.

Step 4 Verify the configuration.

Check the IPv6 routing table of Device A.

```
[~DeviceA] display ipv6 routing-table
Routing Table : _public_
Destinations : 9      Routes : 9

Destination : ::          PrefixLength : 0
NextHop    : 2001:DB8:4::2  Preference   : 60
Cost       : 0            Protocol     : Static
RelayNextHop : ::          TunnelID    : 0x0
Interface   : GigabitEthernet1/0/0  Flags       : D

Destination : 2001:DB8::1  PrefixLength : 128
NextHop    : 2001:DB8::1   Preference   : 0
Cost       : 0            Protocol     : Direct
RelayNextHop : ::          TunnelID    : 0x0
Interface   : InLoopBack0  Flags       : D

Destination : ::FFFF:127.0.0.0  PrefixLength : 104
NextHop    : ::FFFF:127.0.0.1 Preference   : 0
Cost       : 0            Protocol     : Direct
RelayNextHop : ::          TunnelID    : 0x0
Interface   : InLoopBack0  Flags       : D

Destination : ::FFF:127.0.0.1  PrefixLength : 128
NextHop    : 2001:DB8::1   Preference   : 0
Cost       : 0            Protocol     : Direct
RelayNextHop : ::          TunnelID    : 0x0
Interface   : InLoopBack0  Flags       : D

Destination : 2001:DB8:1::  PrefixLength : 64
NextHop    : 2001:DB8:1::1  Preference   : 0
Cost       : 0            Protocol     : Direct
RelayNextHop : ::          TunnelID    : 0x0
Interface   : GigabitEthernet2/0/0  Flags       : D

Destination : 2001:DB8:1::1  PrefixLength : 128
NextHop    : 2001:DB8::1   Preference   : 0
Cost       : 0            Protocol     : Direct
RelayNextHop : ::          TunnelID    : 0x0
Interface   : GigabitEthernet2/0/0  Flags       : D

Destination : 2001:DB8:4::  PrefixLength : 64
NextHop    : 2001:DB8:4::1  Preference   : 0
Cost       : 0            Protocol     : Direct
RelayNextHop : ::          TunnelID    : 0x0
```

```
Interface : GigabitEthernet1/0/0          Flags : D
Destination : 2001:DB8:4::1      PrefixLength : 128
NextHop    : 2001:DB8::1          Preference : 0
Cost       : 0                  Protocol  : Direct
RelayNextHop : ::               TunnelID  : 0x0
Interface   : GigabitEthernet1/0/0          Flags : D

Destination : FE80::          PrefixLength : 10
NextHop    : ::               Preference : 0
Cost       : 0                  Protocol  : Direct
RelayNextHop : ::               TunnelID  : 0x0
Interface   : NULL0             Flags : D
```

Run the **ping** command to verify the connectivity.

```
[~DeviceA] ping ipv6 2001:db8:3::1
PING 2001:DB8:3::1 : 56 data bytes, press CTRL_C to break
Reply from 2001:DB8:3::1
bytes=56 Sequence=1 hop limit=63 time=6 ms
Reply from 2001:DB8:3::1
bytes=56 Sequence=2 hop limit=63 time=2 ms
Reply from 2001:DB8:3::1
bytes=56 Sequence=3 hop limit=63 time=1 ms
Reply from 2001:DB8:3::1
bytes=56 Sequence=4 hop limit=63 time=1 ms
Reply from 2001:DB8:3::1
bytes=56 Sequence=5 hop limit=63 time=1 ms

--- 2001:DB8:3::1 ping statistics---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max=1/2/6 ms
```

Run the **tracert** command to verify the connectivity.

```
[~DeviceA] tracert ipv6 2001:db8:3::1
traceroute to 2001:DB8:3::1 30 hops max,60 bytes packet
1 2001:DB8:4::2 5 ms 1 ms 1 ms
2 2001:DB8:3::1 7 ms 2 ms 3 ms
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
#
ipv6 route-static :: 0 GigabitEthernet 1/0/0 2001:db8:4::2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::2/64
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::2/64
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
#
ipv6 route-static 2001:db8:1:: 64 GigabitEthernet1/0/0 2001:db8:4::1
ipv6 route-static 2001:db8:3:: 64 GigabitEthernet2/0/0 2001:db8:5::1
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::1/64
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
#
ipv6 route-static :: 0 GigabitEthernet1/0/0 2001:db8:5::2
#
return
```

Example for Configuring IPv6 Floating Static Routes

IPv6 Floating static routes can be used for the static route backup.

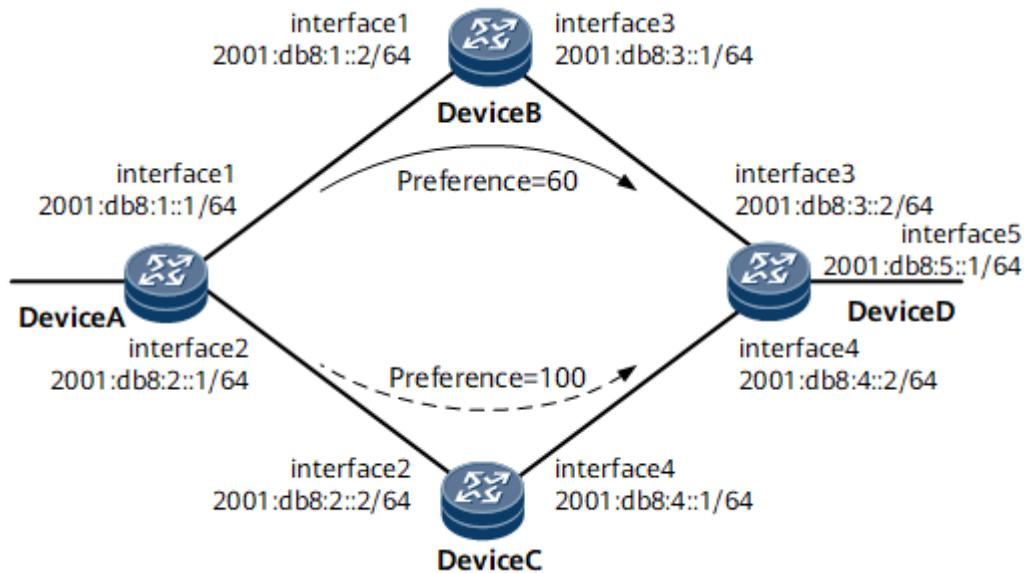
Networking Requirements

Figure 1-53 shows the IPv6 addresses and masks of each router interface and a host. Two IPv6 static routes to 2001:db8:5::/64 are configured on Device A. The primary static route passes through Device B, and the floating static route passes through Device C.

Figure 1-53 Networking for configuring IPv6 floating static routes



Interfaces 1 through 5 in this example represent GE 1/0/1, GE 1/0/2, GE 1/0/3, GE 2/0/3, and GE 3/0/3, respectively.



Precautions

When configuring an IPv6 floating static route, a next-hop address of this route must be specified if the outbound interface is of the broadcast type.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an IPv6 address for each interface of each router.
2. On Devices B and C, configure IPv6 static routes to 2001:db8:5::/64.
3. On Device A, configure two IPv6 static routes to 2001:db8:5::/64 with different priorities.
4. On Device D, configure IPv6 static routes to 2001:db8:1::/64 and 2001:db8:2::/64 so that routers can communicate.

Data Preparation

To complete the configuration, you need the following data:

- On Device A, priority values of two static routes (60 for the one with 2001:db8:1::2 as the next hop address and 100 for the one with 2001:db8:2::2 as the next-hop address)

Procedure

Step 1 Configure an IPv6 address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure IPv6 static routes.

```
# Configure IPv6 static routes on Device B.
```

```
[~DeviceB] ipv6 route-static 2001:db8:5:: 64 2001:db8:3::2
[*DeviceB] commit
```

Configure IPv6 static routes on Device C.

```
[~DeviceC] ipv6 route-static 2001:db8:5:: 64 2001:db8:4::2
[*DeviceC] commit
```

Configure IPv6 static routes on Device A.

```
[*DeviceA] ipv6 route-static 2001:db8:5:: 64 2001:db8:1::2
[*DeviceA] ipv6 route-static 2001:db8:5:: 64 2001:db8:2::2 preference 100
[*DeviceA] commit
```

Configure IPv6 static routes on Device D.

```
[~DeviceD] ipv6 route-static 2001:db8:1:: 64 2001:db8:3::1
[*DeviceD] ipv6 route-static 2001:db8:2:: 64 2001:db8:4::1
[*DeviceD] commit
```

Step 3 Verify the configuration.

View information about static routes in the IP routing table of Device A.

```
[~DeviceA] display ipv6 routing-table protocol static
 _public_ Routing Table : Static
Summary Count : 1

Static routing table status : <Active>
Summary Count : 1

Destination : 2001:db8:5::          PrefixLength : 64
NextHop    : 2001:db8:1::2          Preference   : 60
Cost       : 0                      Protocol     : Static
RelayNextHop : 2001:db8:1::2        TunnelID    : 0x0
Interface   : GigabitEthernet1/0/1   Flags       : RD

Static routing table status : <Inactive>
Summary Count : 0
```

Use the **tracert ipv6** command to check the connectivity on Device A.

```
<DeviceA> tracert ipv6 2001:db8:5::1
traceroute to 2001:db8:5::1 30 hops max,60 bytes packet
1 2001:db8:1::2 195 ms 5 ms 2 ms
2 * 2001:db8:3::2 45 ms !N 5 ms !N
```

Run the **shutdown** command on GE 1/0/1 of Device A to simulate a link fault.

```
[~DeviceA] interface GigabitEthernet 1/0/1
[~DeviceA-GigabitEthernet1/0/1] shutdown
[*DeviceA-GigabitEthernet1/0/1] commit
[*DeviceA-GigabitEthernet1/0/1] quit
[~DeviceA] quit
```

View information about static routes in the IP routing table of Device A. The route to 2001:db8:5::/64 switches to the floating static route with next hop 2001:db8:2::2.

```
<DeviceA> display ipv6 routing-table protocol static
 _public_ Routing Table : Static
Summary Count : 1

Static routing table status : <Active>
Summary Count : 1

Destination : 2001:db8:5::          PrefixLength : 64
NextHop    : 2001:db8:2::2          Preference   : 100
Cost       : 0                      Protocol     : Static
RelayNextHop : 2001:db8:2::2        TunnelID    : 0x0
Interface   : GigabitEthernet1/0/2   Flags       : RD
```

```
Static routing table status : <Inactive>
Summary Count : 0
```

Use the **tracert ipv6** command to check the connectivity on Device A.

```
<DeviceA> tracert ipv6 2001:db8:5::1
traceroute to 2001:db8:5::1 30 hops max,60 bytes packet
1 2001:db8:2::2 87 ms 2 ms 4 ms
2 * 2001:db8:4::2 6 ms !N 2 ms !N
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
#
ipv6 route-static 2001:db8:5:: 64 2001:db8:1::2
ipv6 route-static 2001:db8:5:: 64 2001:db8:2::2 preference 100
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
#
interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
#
ipv6 route-static 2001:db8:5:: 64 2001:db8:3::2
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
#
interface GigabitEthernet2/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
#
ipv6 route-static 2001:db8:5:: 64 2001:db8:4::2
#
return
```

- Device D configuration file

```
#  
sysname DeviceD  
#  
interface GigabitEthernet1/0/3  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:3::2/64  
#  
interface GigabitEthernet2/0/3  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:4::2/64  
#  
interface GigabitEthernet3/0/3  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:5::1/64  
#  
ipv6 route-static 2001:db8:1:: 64 2001:db8:3::1  
ipv6 route-static 2001:db8:2:: 64 2001:db8:4::1  
#  
return
```

Example for Configuring Dynamic BFD for IPv6 Static Routes

Dynamic BFD for IPv6 static routes can fast detect link failures.

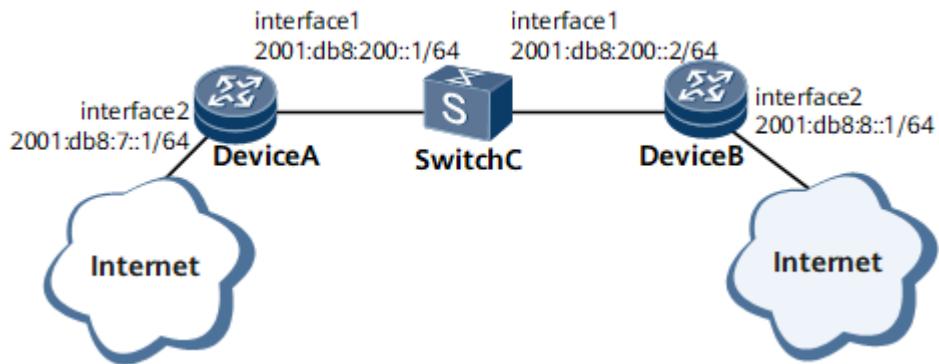
Networking Requirements

On the network shown in [Figure 1-54](#), Device A is connected to Device B through Switch C. A static default route is configured on Device A so that Device A can communicate with external devices. A BFD session is configured between Device A and Device B to detect whether a link fault occurs.

Figure 1-54 Networking for configuring dynamic BFD for IPv6 static routes

NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Precautions

When configuring dynamic BFD for IPv6 static routes, note the following points:

- BFD has been enabled globally.
- The parameters configured on the two ends of a BFD session must be consistent.

Configuration Roadmap

The configuration roadmap is as follows:

1. On Device A, configure an IPv6 static route to Device B.
2. Configure dynamic BFD for IPv6 static routes.

Data Preparation

To complete the configuration, you need the following data:

- Peer IPv6 address to be detected by BFD
- Default values of the local detection multiplier and of the minimum intervals at which BFD Control packets are sent and received

Procedure

Step 1 Configure an IPv6 address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure IPv6 static routes.

On Device A, configure a static route to 2001:db8:8::1/64.

```
[~DeviceA] ipv6 route-static 2001:db8:8:: 64 2001:db8:200::2  
[*DeviceA] commit
```

On Device A, check the IPv6 routing table. The following command output shows that static routes exist in the IPv6 routing table.

```
[~DeviceA] display ipv6 routing-table  
Routing Table : _public_  
Destinations : 6 Routes : 6  
  
Destination : 2001:db8:7:: PrefixLength : 64  
NextHop : 2001:db8:7::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet2/0/0 Flags : D  
  
Destination : 2001:db8:7::1 PrefixLength : 128  
NextHop : ::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet2/0/0 Flags : D  
  
Destination : 2001:db8:8:: PrefixLength : 64  
NextHop : 2001:db8:200::2 Preference : 60  
Cost : 0 Protocol : Static  
RelayNextHop : 2001:db8:200::2 TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : RD  
  
Destination : 2001:db8:200:: PrefixLength : 64  
NextHop : 2001:db8:200::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : D  
  
Destination : 2001:db8:200::1 PrefixLength : 128  
NextHop : ::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : D
```

```
Destination : FE80:: PrefixLength : 10
NextHop   : :: Preference : 0
Cost      : 0 Protocol  : Direct
RelayNextHop : :: TunnelID : 0x0
Interface  : NULL0 Flags    : D
```

On Device B, configure the static route to 2001:db8:7::1/64.

```
[~DeviceB] ipv6 route-static 2001:db8:7:: 64 2001:db8:200::1
[*DeviceB] commit
```

Step 3 Configure dynamic BFD for static routes.

On Device A, bind a static route to a BFD session.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] ipv6 route-static bfd 2001:db8:200::2 local-address 2001:db8:200::1
[*DeviceA] ipv6 route-static 2001:db8:8:: 64 2001:db8:200::2 bfd enable
[*DeviceA] commit
```

On Device B, bind a static route to a BFD session.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] ipv6 route-static bfd 2001:db8:200::1 local-address 2001:db8:200::2
[*DeviceB] ipv6 route-static 2001:db8:7:: 64 2001:db8:200::1 bfd enable
[*DeviceB] commit
```

Step 4 Verify the configuration.

After the configuration is complete, you can view that a BFD session has been established between Device A and Device B and is Up and that a static route is bound to it.

Use the command output on Device A as an example.

```
[~DeviceA] display bfd session all verbose
(w): State in WTR
(*): State is invalid
-----
(Multi Hop) State : Up          Name : dyn_16385
-----
Local Discriminator : 16385      Remote Discriminator : 16385
Session Detect Mode : Asynchronous Mode Without Echo Function
BFD Bind Type      : Peer IP Address
Bind Session Type  : Dynamic
Bind Peer IP Address : 2001:db8:200::2
Bind Interface     : -
Bind Source IP Address : 2001:db8:200::1
FSM Board Id       : 3           TOS-EXP        : 7
Min Tx Interval (ms) : 50        Min Rx Interval (ms) : 50
Actual Tx Interval (ms): 50        Actual Rx Interval (ms): 50
Local Detect Multi : 3           Detect Interval (ms) : 150
Echo Passive       : Disable    Acl Number     : -
Destination Port   : 4784        TTL            : 253
Proc Interface Status : Disable  Process PST    : Disable
WTR Interval (ms)  : 0           Local Demand Mode : Disable
Active Multi       : 3
Last Local Diagnostic : No Diagnostic
Bind Application   : STATICRTV6
Session TX TmrID   : 0           Session Detect TmrID : 0
Session Init TmrID : -           Session WTR TmrID  : -
Session Echo Tx TmrID : -
Session Description : -
Track Group Name   : -
```

Total UP/DOWN Session Number : 1/0

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA  
#  
bfd  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:200::1/64  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:7::1/64  
#  
ipv6 route-static bfd 2001:db8:200::2 local-address 2001:db8:200::1  
ipv6 route-static 2001:db8:8:: 64 2001:db8:200::2 bfd enable  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
bfd  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:200::2/64  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:8::1/64  
#  
ipv6 route-static bfd 2001:db8:200::1 local-address 2001:db8:200::2  
ipv6 route-static 2001:db8:7:: 64 2001:db8:200::1 bfd enable  
#  
return
```

Example for Configuring Static BFD for IPv6 Static Routes

To improve IPv6 network reliability, you can configure static BFD for IPv6 static routes to fast detect link failures and speed up route convergence.

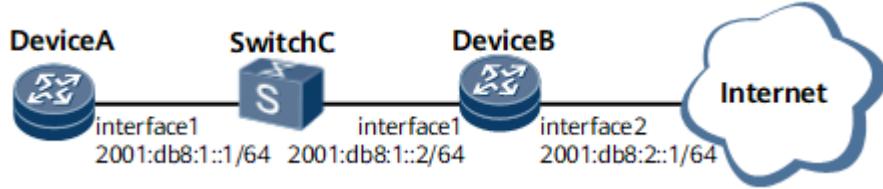
Networking Requirements

On the network shown in [Figure 1-55](#), Device A is connected to Device B through Switch C. A static default route is configured on Device A so that Device A can communicate with external devices. In addition, a BFD session is configured between Device A and Device B to rapidly detect link faults if any.

Figure 1-55 Configuring static BFD for IPv6 static routes

 NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Configure a BFD session on Device A and Device B to detect the link between the two devices.
2. Configure a default static route from Device A to the external network and bind the default static route to a BFD session.

Data Preparation

To complete the configuration, you need the following data:

- Peer IPv6 address to be detected by BFD
- Local discriminator and remote discriminator of a BFD session
- Default values of the local detection multiplier and of the minimum intervals at which BFD Control packets are sent and received

Procedure

Step 1 Configure an IPv6 address for each interface.

For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure a BFD session between Device A and Device B.

On Device A, configure a BFD session between Device A and Device B.

```
<DeviceA> system-view
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bfd aa bind peer-ipv6 2001:db8:1::2
[*DeviceA-bfd-session-aa] discriminator local 10
[*DeviceA-bfd-session-aa] discriminator remote 20
[*DeviceA-bfd-session-aa] commit
[~DeviceA-bfd-session-aa] quit
```

On Device B, configure a BFD session between Device A and Device B.

```
<DeviceB> system-view
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] bfd bb bind peer-ipv6 2001:db8:1::1
[*DeviceB-bfd-session-bb] discriminator local 20
[*DeviceB-bfd-session-bb] discriminator remote 10
```

```
[*DeviceB-bfd-session-bb] commit  
[~DeviceB-bfd-session-bb] quit
```

Step 3 Configure a default static route and bind it to a BFD session.

On Device A, configure a default static route to the external network and bind it to BFD session named aa.

```
[~DeviceA] ipv6 route-static 0::0 0 2001:db8:1::2 track bfd-session aa
```

Step 4 Verify the configuration.

Run the **display bfd session all** command on Device A and Device B. The command output shows that a BFD session has been established and is Up. Then, run the **display current-configuration | include bfd** command in the system view. The command output shows that the default static route has been bound to the BFD session.

Use the command output on Device A as an example.

```
[~DeviceA] display bfd session all  
(w): State in WTR  
(*): State is invalid  
-----  
Local Remote PeerIpAddr State Type InterfaceName  
-----  
10 20 2001:db8:1::2 Up S_IP_PEER -  
-----  
Total UP/DOWN Session Number : 1/0  
[~DeviceA] display current-configuration | include bfd  
bfd  
bfd aa bind peer-ipv6 2001:db8:1::2  
ipv6 route-static :: 0 2001:db8:1::2 track bfd-session aa
```

Check the IP routing table of Device A. The command output shows that the static route exists in the routing table.

```
[~DeviceA] display ipv6 routing-table  
Routing Table : _public_  
Destinations : 5 Routes : 5  
  
Destination : :: PrefixLength : 0  
NextHop : 2001:db8:1::2 Preference : 60  
Cost : 0 Protocol : Static  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : RD  
  
Destination : 2001:db8:3::1 PrefixLength : 128  
NextHop : 2001:db8:3::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : InLoopBack0 Flags : D  
  
Destination : 2001:db8:1:: PrefixLength : 64  
NextHop : 2001:db8:1::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : D  
  
Destination : 2001:db8:1::1 PrefixLength : 128  
NextHop : 2001:db8:3::1 Preference : 0  
Cost : 0 Protocol : Direct  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : D  
  
Destination : FE80:: PrefixLength : 10  
NextHop : :: Preference : 0  
Cost : 0 Protocol : Direct
```

```
RelayNextHop : ::           TunnelID : 0x0
Interface   : NULL0          Flags   : D
```

Run the **shutdown** command on GE 1/0/0 of Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet1/0/0
[~DeviceB-GigabitEthernet1/0/0] shutdown
```

Check the IP routing table of Device A. The command output shows that default route 0::0/0 does not exist. This is because the default static route has been bound to a BFD session. When BFD detects a link fault, BFD rapidly notifies that the bound static route becomes unavailable.

```
[~DeviceA] display ipv6 routing-table
Routing Table : _public_
Destinations : 1      Routes : 1

Destination : 2001:db8:3::1      PrefixLength : 128
NextHop    : 2001:db8:3::1      Preference  : 0
Cost       : 0                  Protocol   : Direct
RelayNextHop : ::              TunnelID   : 0x0
Interface   : InLoopBack0      Flags      : D
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
#
ipv6 route-static :: 0 2001:db8:1::2 track bfd-session aa
#
bfd aa bind peer-ipv6 2001:db8:1::2
discriminator local 10
discriminator remote 20
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
#
bfd bb bind peer-ipv6 2001:db8:1::1
discriminator local 20
discriminator remote 10
#
return
```

Example for Configuring NQA for IPv6 Static Routes

NQA for IPv6 static routes can quickly detect network faults and control the advertisement of static routes.

Networking Requirements

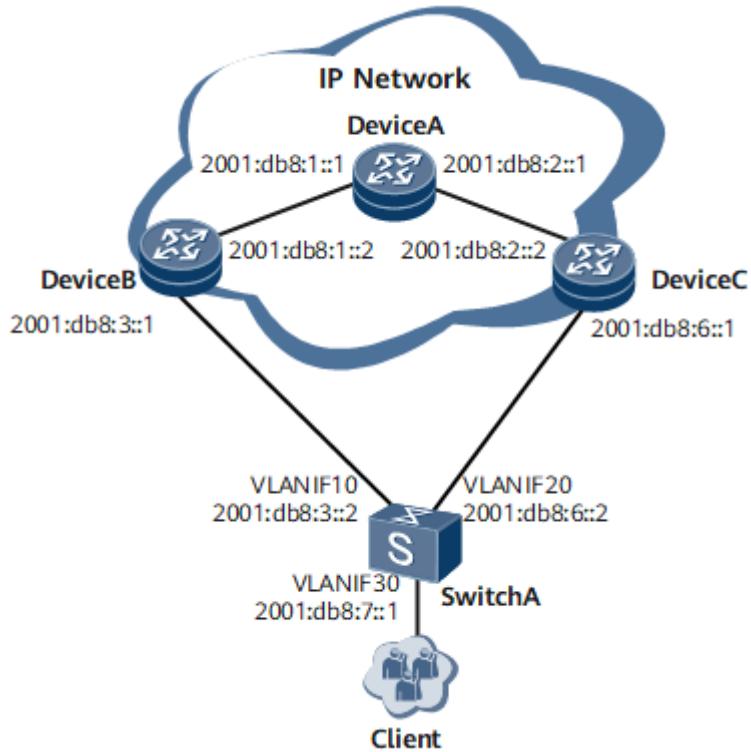
On a simple network or when the router cannot use a dynamic routing protocol to generate routes, you can configure static routes. Unlike dynamic routing protocols, static routes do not have a detection mechanism. If a link fails, a network administrator must manually delete the corresponding static route from the IPv6 routing table, which delays link switchovers and causes a lengthy service interruption.

Bidirectional Forwarding Detection (BFD) for static routes is adaptable to link changes but requires that both ends of a link support BFD. If either end of a link does not support BFD, configure NQA for IPv6 static routes. If an NQA test instance detects a link fault, it instructs the routing management module to delete the associated static route from the IPv6 routing table. Then traffic is switched to a backup route to prevent lengthy service interruptions.

In [Figure 1-56](#), backup links are deployed on the IP metropolitan area network (MAN).

- Static routes are configured on Device B and Device C. Device B is the master device, while Device C is the backup device.
- In most cases, traffic is transmitted over the primary link (Device B -> Switch A).
- If the primary link fails, traffic switches to the backup link (Device C -> Switch A).

Figure 1-56 NQA for IPv6 static routes



Device Name	Interface	IP Address
DeviceA	GE 1/0/0	2001:db8:1::1/64
	GE 2/0/3	2001:db8:2::1/64
DeviceB	GE 1/0/0	2001:db8:1::2/64
	GE 1/0/1	2001:db8:3::1/64
DeviceC	GE 1/0/0	2001:db8:6::1/64
	GE 2/0/3	2001:db8:2::2/64
SwitchA	VLANIF 10	2001:db8:3::2/64
	VLANIF 20	2001:db8:6::2/64
	VLANIF 30	2001:db8:7::1/64

NOTE

In this networking, Switch A provides access services for users. In actual networking, optical line terminals (OLTs), digital subscriber line access multiplexers (DSLAMs), multiservice access nodes (MSANs), or x digital subscriber lines (xDSLs) can be used for user access. The configurations on Device A, Device B, and Device C are the same.

Configuration Roadmap

The configuration roadmap is as follows:

1. Create an Internet Control Message Protocol (ICMP) NQA test instance to monitor the status of the primary link.
Create an ICMP NQA test instance on the NQA client Device B to test whether the primary link is running properly.
2. Configure static routes and associate the static route along the primary link with the ICMP NQA test instance.
Configure static routes on Device B and Device C, and associate the static route configured on Device B with the ICMP NQA test instance. If the ICMP NQA test instance detects a link fault, it instructs the routing management module to delete the associated static route from the IPv6 routing table.
3. Configure a dynamic routing protocol.
Configure a dynamic routing protocol on Device A, Device B, and Device C so that they can learn routes from one another.
4. Configure the dynamic routing protocol to import static routes and set a higher cost for the static route along the backup link than for the one along the primary link.
Configure the dynamic routing protocol on Device B and Device C to import static routes, and set a higher cost for the static route imported by Device C than for the one imported by Device B. This configuration allows Device A to preferentially select the link (Device B -> Switch A).

Data Preparation

To complete the configuration, you need the following data:

- Interface IPv6 addresses
- NQA item values (for details, see [Table 1-13](#))

Table 1-13 NQA item values

Item	Value
Administrator name	admin
Name of the test instance	test
Test type	ICMP
Destination address	2001:db8:3::2
Interval at which the NQA test automatically runs	3 seconds

- OSPFv3 backbone area (Area 0) of Device A, Device B, and Device C, and their router IDs (1.1.1.1, 2.2.2.2, and 3.3.3.3)

Procedure

Step 1 Configure interface IPv6 addresses. For configuration details, see [Configuration Files](#) in this section.

Step 2 Create an NQA test instance on Device B to test the link between Device B and Switch A.

```
<DeviceB> system-view
[~DeviceB] nqa test-instance admin test
[*DeviceB-nqa-admin-test] test-type icmp
[*DeviceB-nqa-admin-test] destination-address ipv6 2001:db8:3::2
[*DeviceB-nqa-admin-test] frequency 9
[*DeviceB-nqa-admin-test] interval seconds 3
[*DeviceB-nqa-admin-test] start now
[*DeviceB-nqa-admin-test] commit
[~DeviceB-nqa-admin-test] quit
```

Step 3 Configure IPv6 static routes.

```
# Configure an IPv6 static route on Device B and associate it with the NQA test instance.
```

```
[~DeviceB] ipv6 route-static 2001:db8:7:: 64 GigabitEthernet 1/0/1 2001:db8:3::2 track nqa admin test
[*DeviceB] commit
```

```
# Configure an IPv6 static route on Device C.
```

```
[*DeviceC] ipv6 route-static 2001:db8:7:: 64 GigabitEthernet 1/0/0 2001:db8:6::2
[*DeviceC] commit
```

NOTE

The next hop address of the IPv6 static route configured on the local end must be the link-local address of the peer end, which can be obtained using the **display ipv6 interface [interface-type interface-number]** command on the peer end.

Step 4 Configure a dynamic routing protocol on Device A, Device B, and Device C.

Configure OSPFv3 on Device A.

```
<DeviceA> system-view
[~DeviceA] interface GigabitEthernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] interface GigabitEthernet 2/0/3
[*DeviceA-GigabitEthernet2/0/3] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet2/0/3] commit
[~DeviceA-GigabitEthernet2/0/3] quit
```

Configure OSPFv3 on Device B.

```
[~DeviceB] interface GigabitEthernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

Configure OSPFv3 on Device C.

```
[~DeviceC] interface GigabitEthernet 2/0/3
[~DeviceC-GigabitEthernet2/0/3] ospfv3 1 area 0.0.0.0
[*DeviceC-GigabitEthernet2/0/3] commit
[~DeviceC-GigabitEthernet2/0/3] quit
```

Step 5 Configure OSPFv3 on Device B and Device C to import static routes.

Configure OSPFv3 on Device B to import a static route and set the cost to 10 for the static route.

```
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] import-route static cost 10
[*DeviceB-ospfv3-1] commit
[~DeviceB-ospfv3-1] quit
```

Configure OSPFv3 on Device C to import a static route and set the cost to 20 for the static route.

```
[*DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] import-route static cost 20
[*DeviceC-ospfv3-1] commit
[~DeviceC-ospfv3-1] quit
```

Step 6 Verify the configuration.

After the configuration is complete, run the **display current-configuration | include nqa** command on Device B in the system view. The command output shows that the IPv6 static route has been associated with the NQA test instance. Run the **display nqa results** command. The command output shows that an NQA test instance has been created.

Display configurations of NQA for IPv6 static routes.

```
[~DeviceB] display current-configuration | include nqa
ipv6 route-static 2001:db8:7:: 64 GigabitEthernet1/0/1 2001:db8:3::2 track nqa admin test
nqa test-instance admin test
```

Display NQA test results.

```
[~DeviceB] display nqa results test-instance admin test
NQA entry(admin, test) : testflag is active ,testtype is icmp
1 . Test 359 result    The test is finished
Send operation times: 3          Receive response times: 3
Completion:success           RTD OverThresholds number:0
Attempts number:1            Drop operation number:0
```

```
Disconnect operation number:0      Operation timeout number:0
System busy operation number:0    Connection fail number:0
Operation sequence errors number:0 RTT Stats errors number:0
Destination ip address:2001:db8:3::2
Min/Max/Average Completion Time: 1/3/2
Sum/Square-Sum Completion Time: 7/19
Last Good Probe Time: 2012-11-14 12:15:22.3
Lost packet ratio: 0 %
```

The command output shows "Lost packet ratio 0 %," indicating that the link is running properly.

Display the routing table on Device B.

```
[~DeviceB] display ipv6 routing-table 2001:db8:7::
Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:7::          PrefixLength : 64
NextHop    : 2001:db8:3::2        Preference   : 60
Cost       : 0                   Protocol     : Static
RelayNextHop : ::                TunnelID    : 0x0
Interface   : GigabitEthernet1/0/1 Flags       : D
```

The command output shows that the static route exists in the routing table.

Display the routing table on Device A.

```
[~DeviceA] display ipv6 routing-table 2001:db8:7::
Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:7::          PrefixLength : 64
NextHop    : FE80::2200:10FF:FE03:0  Preference   : 150
Cost       : 10                  Protocol     : OSPFv3ASE
RelayNextHop : ::                TunnelID    : 0x0
Interface   : GigabitEthernet1/0/0 Flags       : D
```

The command output shows that a route to 2001:db8:7::1/128 exists in the routing table. The outbound interface of the route is GE 1/0/0 and the cost is 10. Traffic is preferentially transmitted along the link Device B -> Switch A.

Shut down GE 1/0/1 on Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/1
[*DeviceB-GigabitEthernet1/0/1] shutdown
[*DeviceB-GigabitEthernet1/0/1] commit
[~DeviceB-GigabitEthernet1/0/1] quit
```

Display NQA test results.

```
[~DeviceB] display nqa results test-instance admin test
NQA entry(admin, test) : testflag is active ,testtype is icmp
1 . Test 1156 result  The test is finished
Send operation times: 3      Receive response times: 0
Completion:failed          RTD OverThresholds number:0
Attempts number:1           Drop operation number:0
Disconnect operation number:0 Operation timeout number:3
System busy operation number:0 Connection fail number:0
Operation sequence errors number:0 RTT Stats errors number:0
Destination ip address:2001:db8:3::2
Min/Max/Average Completion Time: 0/0/0
Sum/Square-Sum Completion Time: 0/0
Last Good Probe Time: 0000-00-00 00:00:00.0
Lost packet ratio: 100 %
```

The command output shows "Completion:failed" and "Lost packet ratio is 100 %," indicating that the link is faulty.

Display the routing table on Device B.

```
[~DeviceB] display ipv6 routing-table 2001:db8:7::  
Routing Table : _public_  
Summary Count : 1  
  
Destination : 2001:db8:7:: PrefixLength : 64  
NextHop : FE80::3A00:10FF:FE03:0 Cost : 20 Preference : 150  
Protocol : OSPFv3ASE  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet1/0/0 Flags : D
```

The command output shows that the static route has been deleted and that the route has become an OSPFv3 route learned from Device A.

Display the routing table on Device A.

```
[~DeviceA] display ipv6 routing-table 2001:db8:7::  
Routing Table : _public_  
Summary Count : 1  
  
Destination : 2001:db8:7:: PrefixLength : 64  
NextHop : FE80::3A00:10FF:FE03:107 Cost : 20 Preference : 150  
Protocol : OSPFv3ASE  
RelayNextHop : :: TunnelID : 0x0  
Interface : GigabitEthernet2/0/3 Flags : D
```

The static route has been associated with the NQA test instance on Device B. If NQA detects a link fault, it rapidly notifies Device B that the associated static route is unavailable. Device A cannot learn the route to 2001:db8:7::/64 from Device B. However, Device A can learn the route to 2001:db8:7::/64 from Device C. The route's outbound interface is GE 2/0/3, and the cost is 20. Traffic switches to the link Device C -> Switch A.

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA  
#  
ospfv3 1  
router-id 1.1.1.1  
area 0.0.0.  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:1::1/64  
ospfv3 1 area 0.0.0.0  
#  
interface GigabitEthernet2/0/3  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:2::1/64  
ospfv3 1 area 0.0.0.0  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
ospfv3 1  
router-id 2.2.2.2
```

```
import-route static cost 10
area 0.0.0.
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
ospfv3 1 area 0.0.0.
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
#
interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::1/64
#
ipv6 route-static 2001:db8:7:: 64 GigabitEthernet1/0/1 2001:db8:3::2 track nqa admin test
#
nqa test-instance admin test
test-type icmp
destination-address ipv6 2001:db8:3::2
interval seconds 3
frequency 9
start now
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
ospfv3 1
router-id 3.3.3.3
import-route static cost 20
area 0.0.0.
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:6::1/64
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
#
interface GigabitEthernet2/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
ospfv3 1 area 0.0.0.
#
ipv6 route-static 2001:db8:7:: 64 GigabitEthernet1/0/0 2001:db8:6::2
#
return
```

Example for Configuring NQA Group for IPv6 Static Route

NQA group for IPv6 static route allows an NQA group to be associated with multiple NQA test instances to monitor multiple links, rapidly detect network faults, and control advertisement of IPv6 static routes, thereby implementing service switching.

Networking Requirements

NQA group for IPv6 static route monitors the link status of an IPv6 static route through association between the static route and an NQA group that consists of multiple NQA test instances. The RM module determines whether the IPv6 static route is active based on the test result of the NQA group. If the IPv6 static route is inactive, the RM module deletes it from the IP routing table and selects an available backup link for data forwarding, which prevents lengthy service interruptions.

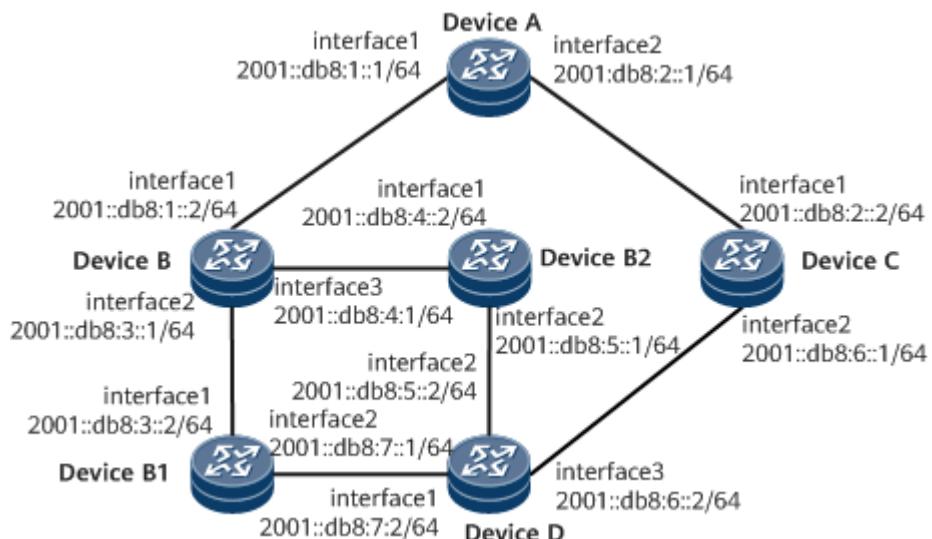
Figure 1-57 shows a network with redundant links.

- BGP peer relationships are established between Device A and Device B and between Device A and Device C.
- On Device B and Device C, default static routes are configured and imported into BGP, and different PrefVal values are configured so that Device B and Device C function as the master device and backup device, respectively.
- Device B and Device D are connected through two links by way of Device B1 and Device B2.
- In normal cases, traffic is transmitted along the primary link (Device B -> Device D).
- If the primary link fails (the NQA group bound to the IPv6 default static route of Device B considers the IPv6 static route inactive when monitoring the two links from Device B to Device D), traffic is switched to the backup link (Device C -> Device D).

Figure 1-57 Network diagram of NQA group for IPv6 static route

NOTE

In this example, interface1, interface2, and interface3 represent GE 1/0/0, GE 1/0/1, and GE 1/0/2, respectively.



Configuration Roadmap

The configuration roadmap is as follows:

1. Create NQA test instances of the ICMP type to detect faults on the primary link.

Create two NQA test instances of the ICMP type between Device B and Device B1 and between Device B and Device B2 to correspond to the two links between Device B and Device D and to check whether the primary link (Device B -> Device D) is running properly.
2. Create an NQA group.

Bind the two NQA test instances of the ICMP type to the NQA group.
3. Configure static routes and associate the static route along the primary link with the NQA group.

Configure IPv6 static routes to loopback0 interfaces between Device A and Device B, and between Device A and Device C. Configure default IPv6 static routes on Device B and Device C. Associate the IPv6 static route configured on Device B with the NQA group, which collects the detection results of two NQA test instances to determine its status. When the NQA group goes down, the device determines that the current link is faulty and instructs the RM module to delete the IPv6 static route from the IP routing table.
4. Configure BGP.

Configure BGP on Device A, Device B, and Device C so that they can learn routes from each other.
5. Configure BGP to import IPv6 static routes and set a larger PrefVal value for the primary link.

Configure BGP on Device B and Device C to import IPv6 static routes, and set a larger PrefVal value for the IPv6 static route imported by Device B. When Device A learns routes to the same destination from Device B and Device C, it preferentially selects the link (Device B -> Device D), which has a larger PrefVal value.

Data Preparation

To complete the configuration, you need the following data:

- IP address of each interface
- NQA-related configurations (for details, see [Table 1-14](#))

Table 1-14 NQA parameter values

Item	Value
Administrator name	user1, user2
Test instance name	test1, test2
Test type	ICMP
Destination address	2001::db8:3::2/64, 2001::db8:4::2/64
Test interval	10s
Number of probes	2
Interval at which packets are sent	5s

Item	Value
Timeout period	4s

- Operation type between test instances in the NQA group: OR
- BGP AS numbers (the same) and router IDs of Device A, Device B, and Device C (1.1.1.1, 2.2.2.2, and 3.3.3.3, respectively)

Procedure

Step 1 Configure IP addresses. For details, see [Configuration Files](#).

Step 2 On Device B, configure two NQA test instances of the ICMP type between Device B and Device B1 and between Device B and Device B2.

```
<DeviceB> system-view
[~DeviceB] nqa test-instance user1 test1
[*DeviceB-nqa-user1-test1] test-type icmp
[*DeviceB-nqa-user1-test1] destination-address ipv6 2001:db8:3::2
[*DeviceB-nqa-user1-test1] frequency 10
[*DeviceB-nqa-user1-test1] probe-count 2
[*DeviceB-nqa-user1-test1] interval seconds 5
[*DeviceB-nqa-user1-test1] timeout 4
[*DeviceB-nqa-user1-test1] start now
[*DeviceB-nqa-user1-test1] commit
[~DeviceB-nqa-user1-test1] quit
[~DeviceB] nqa test-instance user2 test2
[*DeviceB-nqa-user2-test2] test-type icmp
[*DeviceB-nqa-user2-test2] destination-address ipv6 2001:db8:4::2
[*DeviceB-nqa-user2-test2] frequency 10
[*DeviceB-nqa-user2-test2] probe-count 2
[*DeviceB-nqa-user2-test2] interval seconds 5
[*DeviceB-nqa-user2-test2] timeout 4
[*DeviceB-nqa-user2-test2] start now
[*DeviceB-nqa-user2-test2] commit
[~DeviceB-nqa-user2-test2] quit
```

Step 3 Create an NQA group and bind the two NQA test instances of the ICMP type to the group.

```
[~DeviceB] nqa group group1
[*DeviceB-nqa-group-group1] nqa test-instance user1 test1
[*DeviceB-nqa-group-group1] nqa test-instance user2 test2
[*DeviceB-nqa-group-group1] operator or
[*DeviceB-nqa-group-group1] commit
[~DeviceB-nqa-group-group1] quit
```

Step 4 Configure static routes.

```
# Configure static routes to loopback interfaces between Device A and Device B,
and between Device A and Device C.
```

```
[~DeviceA] ipv6 route-static 2::2 128 GigabitEthernet 1/0/0 2001:db8:1::2
[~DeviceA] ipv6 route-static 3::3 128 GigabitEthernet 1/0/1 2001:db8:2::2
[*DeviceA] commit
```

```
# Configure a static route to the loopback interface of Device A on Device B.
```

```
[~DeviceB] ipv6 route-static 1::1 128 GigabitEthernet 1/0/0 2001:db8:1::1
[*DeviceB] commit
```

```
# Configure a static route to the loopback interface of Device A on Device C.
```

```
[~DeviceC] ipv6 route-static 1::1 128 GigabitEthernet 1/0/1 2001:db8:2::1
[*DeviceC] commit
```

Configure a default static route on Device B and associate it with the NQA group.

```
[*DeviceB] ipv6 route-static :: 128 NULL0 track nqa-group group1  
[*DeviceB] commit
```

Configure a default static route on Device C.

```
[*DeviceC] ipv6 route-static :: 128 NULL0  
[*DeviceC] commit
```

Step 5 Configure BGP on Device A, Device B, and Device C.

Configure BGP on Device A.

```
[~DeviceA] bgp 100  
[*DeviceA-bgp] peer 2::2 as-number 100  
[*DeviceA-bgp] peer 2::2 connect-interface LoopBack0  
[*DeviceA-bgp] peer 3::3 as-number 100  
[*DeviceA-bgp] peer 3::3 connect-interface LoopBack0  
[*DeviceA-bgp] quit  
[*DeviceA] commit
```

Configure BGP on Device B.

```
[~DeviceB] bgp 100  
[*DeviceB-bgp] peer 1::1 as-number 100  
[*DeviceB-bgp] peer 1::1 connect-interface LoopBack0  
[*DeviceB-bgp] quit  
[*DeviceB] commit
```

Configure BGP on Device C.

```
[~DeviceC] bgp 100  
[*DeviceC-bgp] peer 1::1 as-number 100  
[*DeviceC-bgp] peer 1::1 connect-interface LoopBack0  
[*DeviceC-bgp] quit  
[*DeviceC] commit
```

Step 6 Configure BGP on Device B and Device C to import IPv6 static routes, and set a larger PrefVal value for the primary link.

Configure BGP on Device B to import IPv6 static routes, and set the PrefVal value to 200.

```
[~DeviceB] bgp 100  
[*DeviceB-bgp] import-route static  
[*DeviceB-bgp] ipv6-family unicast  
[*DeviceB-bgp-af-ipv6] peer 1::1 preferred-value 200  
[*DeviceB-bgp-af-ipv6] commit  
[~DeviceB-bgp-af-ipv6] quit  
[~DeviceB-bgp] quit
```

Configure BGP on Device C to import IPv6 static routes, and set the PrefVal value to 100.

```
[~DeviceC] bgp 100  
[*DeviceC-bgp] import-route static  
[*DeviceC-bgp] ipv6-family unicast  
[*DeviceC-bgp-af-ipv6] peer 1::1 preferred-value 100  
[*DeviceC-bgp-af-ipv6] commit  
[~DeviceC-bgp-af-ipv6] quit  
[~DeviceC-bgp] quit
```

Step 7 Verify the configuration.

Check the results of the two NQA test instances on Device B.

```
[~DeviceB] display nqa results test-instance user1 test1
```

```
NQA entry(user1, test1) :testflag is inactive ,testtype is icmp
1 . Test 1 result The test is finished
Send operation times: 3          Receive response times: 3
Completion:success           RTD OverThresholds number:0
Attempts number:1              Drop operation number:0
Disconnect operation number:0   Operation timeout number:0
System busy operation number:0  Connection fail number:0
Operation sequence errors number:0  RTT Status errors number:0
Destination ip address:172.16.1.2
Min/Max/Average Completion Time: 11/185/69
Sum/Square-Sum Completion Time: 207/34467
Last Good Probe Time: 2022-09-24 15:08:38.6
Lost packet ratio: 0 %
[~DeviceB] display nqa results test-instance user2 test2
NQA entry(user2, test2) :testflag is inactive ,testtype is icmp
1 . Test 1 result The test is finished
Send operation times: 3          Receive response times: 3
Completion:success           RTD OverThresholds number:0
Attempts number:1              Drop operation number:0
Disconnect operation number:0   Operation timeout number:0
System busy operation number:0  Connection fail number:0
Operation sequence errors number:0  RTT Status errors number:0
Destination ip address:172.16.2.2
Min/Max/Average Completion Time: 9/16/11
Sum/Square-Sum Completion Time: 34/418
Last Good Probe Time: 2022-09-24 15:08:50.3
Lost packet ratio: 0 %
```

The command output shows "Lost packet ratio 0 %," indicating that the link is running properly.

Check the test result of the NQA group on Device B.

```
[~DeviceB] display nqa group
NQA-group information:
-----
NQA-group group1
Status: UP      Operator: OR
-----
Admin-name        Test-name        Status
-----
user1            test1           UP
user2            test2           UP
```

Check the routing table of Device A.

```
[~DeviceA] display ipv6 routing-table | include IBGP
Routing Table : _public_
Summary Count : 1

Destination : ::          PrefixLength : 128
NextHop    : 2::2          Preference  : 255
Cost       : 0             Protocol   : IBGP
RelayNextHop : ::          TunnelID   : 0x0
Interface   : GigabitEthernet1/0/0     Flags      : RD
```

The command output shows that the next hop used to forward user traffic on Device A is Device B, indicating that user traffic is transmitted along the primary link.

Shut down GigabitEthernet 1/0/1 on Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/1
[~DeviceB-GigabitEthernet1/0/1] shutdown
[*DeviceB-GigabitEthernet1/0/1] commit
[~DeviceB] quit
```

Check the NQA test result.

```
[~DeviceB] display nqa results test-instance user1 test1
NQA entry(user1, test1) :testflag is inactive ,testtype is icmp
2 . Test 2 result The test is finished
Send operation times: 3           Receive response times: 0
Completion:failed           RTD OverThresholds number:0
Attempts number:1           Drop operation number:0
Disconnect operation number:0     Operation timeout number:3
System busy operation number:0    Connection fail number:0
Operation sequence errors number:0   RTT Status errors number:0
Destination ip address:172.16.1.2
Min/Max/Average Completion Time: 0/0/0
Sum/Square-Sum Completion Time: 0/0
Last Good Probe Time: 0000-00-00 00:00:00.0
Lost packet ratio: 100 %
```

"Completion:failed" and "Lost packet ratio: 100%" are displayed, indicating that the link where GigabitEthernet 1/0/1 resides is faulty.

Check the test result of the NQA group on Device B.

```
[~DeviceB] display nqa group
NQA-group information:
-----
NQA-group group1
Status: UP      Operator: OR
-----
Admin-name          Test-name        Status
-----
user1              test1           DOWN
user2              test2           UP
```

The command output shows that the status of the NQA group remains UP.

Check the routing table of Device A.

```
[~DeviceA] display ipv6 routing-table | include IBGP
Routing Table : _public_
Summary Count : 1
-----
Destination : ::          PrefixLength : 128
NextHop    : 2::2          Preference  : 255
Cost       : 0             Protocol   : IBGP
RelayNextHop : ::          TunnelID   : 0x0
Interface  : GigabitEthernet1/0/0   Flags     : RD
```

The operation type between test instances in the NQA group is OR. Therefore, the status of the NQA group remains UP.

Shut down GigabitEthernet 1/0/2 on Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 1/0/2
[~DeviceB-GigabitEthernet1/0/2] shutdown
[*DeviceB-GigabitEthernet1/0/2] commit
[~DeviceB] quit
```

Check the NQA test result.

```
[~DeviceB] display nqa results test-instance user2 test2
NQA entry(user2, test2) : testflag is active ,testtype is icmp
1 . Test 186 result The test is finished
Send operation times: 2           Receive response times: 0
Completion:failed           RTD OverThresholds number:0
Attempts number:1           Drop operation number:0
Disconnect operation number:0     Operation timeout number:2
System busy operation number:0    Connection fail number:0
Operation sequence errors number:0   RTT Stats errors number:0
Destination ip address:172.16.2.2
Min/Max/Average Completion Time: 0/0/0
Sum/Square-Sum Completion Time: 0/0
```

```
Last Good Probe Time: 0000-00-00 00:00:00.0
Lost packet ratio: 100 %
```

"Completion:failed" and "Lost packet ratio: 100%" are displayed, indicating that the link where GigabitEthernet 1/0/2 resides is faulty.

Check the test result of the NQA group on Device B.

```
[~DeviceB] display nqa group
NQA-group information:
-----
NQA-group group1
Status: DOWN      Operator: OR
-----
Admin-name          Test-name        Status
-----
user1              test1           DOWN
user2              test2           DOWN
```

The command output shows that the status of the NQA group changes to DOWN.

Check the routing table of Device A.

```
[~DeviceA] display ipv6 routing-table | include IBGP
Routing Table : _public_
Summary Count : 1
-----
Destination : ::          PrefixLength : 128
NextHop    : 3::3          Preference   : 255
Cost       : 0             Protocol     : IBGP
RelayNextHop : ::          TunnelID    : 0x0
Interface   : GigabitEthernet1/0/1   Flags       : RD
```

The command output shows that the next hop used to forward user traffic on Device A changes to Device C, indicating that the traffic is switched to the backup link.

The NQA group on Device B is associated with the IPv6 static route. When the NQA group detects that both links from Device B to Device D fail, the NQA group goes down and rapidly notifies Device B that the associated IPv6 static route is unavailable. Service traffic is then switched to the backup link.

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
interface LoopBack0
ipv6 enable
ipv6 address 1::1/128
#
interface GigabitEthernet1/0/0
ipv6 enable
ipv6 address 2001:DB8:1::1/64
#
interface GigabitEthernet1/0/1
ipv6 enable
ipv6 address 2001:DB8:2::1/64
#
bgp 100
peer 2::2 as-number 100
```

```
peer 2::2 connect-interface LoopBack0
peer 3::3 as-number 100
peer 3::3 connect-interface LoopBack0
#
ipv6-family unicast
undo synchronization
peer 2::2 enable
peer 3::3 enable
#
ipv6 route-static 2::2 128 GigabitEthernet1/0/0 2001:DB8:1::2
ipv6 route-static 3::3 128 GigabitEthernet1/0/1 2001:DB8:2::2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
router id 2.2.2.2
#
interface LoopBack0
    ipv6 enable
    ipv6 address 2::2/128
#
interface GigabitEthernet1/0/0
    ipv6 enable
    ipv6 address 2001:DB8:1::2/64
#
interface GigabitEthernet1/0/1
    ipv6 enable
    ipv6 address 2001:DB8:3::1/64
#
interface GigabitEthernet1/0/2
    ipv6 enable
    ipv6 address 2001:DB8:4::1/64
#
bgp 100
    peer 1::1 as-number 100
    peer 1::1 connect-interface LoopBack0
    #
    ipv6-family unicast
        import-route static
        peer 1::1 preferred-value 200
    #
    ipv6 route-static :: 128 NULL0 track nqa-group group1
    ipv6 route-static 1::1 128 GigabitEthernet1/0/0 2001:DB8:1::1
    #
    nqa test-instance user1 test1
        test-type icmp
        destination-address ipv6 2001:DB8:3::1
        interval seconds 5
        timeout 4
        probe-count 2
        frequency 10
        start now
    #
    nqa test-instance user2 test2
        test-type icmp
        destination-address ipv6 2001:DB8:4::1
        interval seconds 5
        timeout 4
        probe-count 2
        frequency 10
        start now
    #
    nqa group group1
        nqa test-instance user1 test1
        nqa test-instance user2 test2
        operator or
```

```
#  
return  
● Device C configuration file  
#  
sysname DeviceC  
#  
router id 3.3.3.3  
#  
interface LoopBack0  
ipv6 enable  
ipv6 address 3::3/128  
#  
interface GigabitEthernet1/0/0  
ipv6 enable  
ipv6 address 2001:DB8:6::1/64  
#  
interface GigabitEthernet1/0/1  
ipv6 enable  
ipv6 address 2001:DB8:2::2/64  
#  
bgp 100  
peer 1::1 as-number 100  
peer 1::1 connect-interface LoopBack0  
#  
ipv6-family unicast  
import-route static  
peer 1::1 preferred-value 100  
#  
ipv6 route-static :: 128 NULL0  
ipv6 route-static 1::1 128 GigabitEthernet1/0/1 2001:DB8:2::1  
#  
return
```

Example for Configuring FRR for IPv6 Static Routes on the Public Network

FRR for IPv6 static routes on the public network can fast detect link failures.

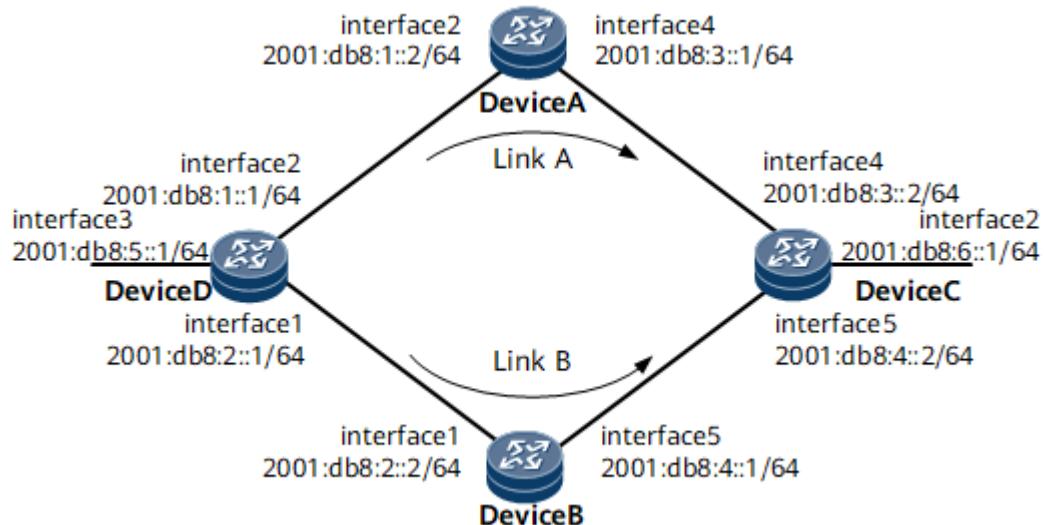
Networking Requirements

On the network shown in [Figure 1-58](#), it is required that two IPv6 static routes with Device A and Device B as the next hops be configured on Device D and that Link B function as the backup of Link A. If Link A fails, traffic is switched to the Link B immediately.

Figure 1-58 Networking for configuring FRR for IPv6 static routes on the public network



In this example, interface1, interface2, interface3, interface4, and interface5 represent GE 1/0/1, GE 1/0/2, GE 1/0/3, GE 2/0/3, and GE 3/0/3, respectively.



Precautions

When configuring FRR for IPv6 static routes on the public network, ensure that there are at least two IPv6 static routes to the same destination address.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure two IPv6 static routes with Device A and Device B as the next hops on Device D.
2. On Device D, set a higher priority for Link A to ensure that Link A becomes the primary link.
3. Enable FRR for IPv6 static routes on Device D, and check the backup outbound interface and the backup next hop.
4. Configure static BFD for IPv6 static routes to speed up fault detection.

NOTE

To speed up fault detection, configure dynamic or static BFD for IPv6 static routes. In this example, static BFD for IPv6 static routes is used. This is because it is more commonly used on the live network.

5. Disable FRR for IPv6 static routes, and check the backup outbound interface and the backup next hop.

Data Preparation

To complete the configuration, you need preference values of IPv6 static routes.

Procedure

- Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure IPv6 static routes.

On Device A, configure IPv6 static routes.

```
[~DeviceA] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet1/0/2 2001:db8:1::1
[*DeviceA] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet2/0/3 2001:db8:3::2
[*DeviceA] commit
```

On Device B, configure static routes.

```
[~DeviceB] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet1/0/1 2001:db8:2::1
[*DeviceB] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet3/0/3 2001:db8:4::2
[*DeviceB] commit
```

On Device C, configure IPv6 static routes.

```
[~DeviceC] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet2/0/3 2001:db8:3::1
[*DeviceC] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet3/0/3 2001:db8:4::1
[*DeviceC] ipv6 route-static 2001:db8:1:: 64 GigabitEthernet2/0/3 2001:db8:3::1
[*DeviceC] ipv6 route-static 2001:db8:2:: 64 GigabitEthernet3/0/3 2001:db8:4::1
[*DeviceC] commit
```

On Device D, configure IPv6 static routes.

```
[~DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet1/0/2 2001:db8:1::2
[*DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet1/0/1 2001:db8:2::2
[*DeviceD] ipv6 route-static 2001:db8:3:: 64 GigabitEthernet1/0/2 2001:db8:1::2
[*DeviceD] ipv6 route-static 2001:db8:4:: 64 GigabitEthernet1/0/1 2001:db8:2::2
[*DeviceD] commit
[~DeviceD] quit
```

Check the IP routing table of Device D. The following command output shows that load balancing is performed between the two IPv6 static routes.

```
<DeviceD> display ipv6 routing-table
Routing Table : _public_
Destinations : 13      Routes : 13

Destination : ::1          PrefixLength : 128
NextHop    : ::1           Preference  : 0
Cost       : 0             Protocol   : Direct
RelayNextHop : ::           TunnelID   : 0x0
Interface  : InLoopBack0  Flags       : D

Destination : ::FFFF:127.0.0.0  PrefixLength : 104
NextHop    : ::FFFF:127.0.0.1 Preference  : 0
Cost       : 0             Protocol   : Direct
RelayNextHop : ::           TunnelID   : 0x0
Interface  : InLoopBack0  Flags       : D

Destination : ::FFFF:127.0.0.1  PrefixLength : 128
NextHop    : ::1           Preference  : 0
Cost       : 0             Protocol   : Direct
RelayNextHop : ::           TunnelID   : 0x0
Interface  : InLoopBack0  Flags       : D

Destination : 2001:db8:1::  PrefixLength : 64
NextHop    : 2001:db8:1::1 Preference  : 0
Cost       : 0             Protocol   : Direct
RelayNextHop : ::           TunnelID   : 0x0
Interface  : GigabitEthernet1/0/2 Flags       : D

Destination : 2001:db8:1::1  PrefixLength : 128
NextHop    : ::1           Preference  : 0
Cost       : 0             Protocol   : Direct
RelayNextHop : ::           TunnelID   : 0x0
Interface  : GigabitEthernet1/0/2 Flags       : D

Destination : 2001:DB8:2::  PrefixLength : 64
```

NextHop : 2001:DB8:2::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:DB8:2::1	PrefixLength : 128
NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:DB8:3::	PrefixLength : 64
NextHop : 2001:DB8:1::2	Preference : 60
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/2	Flags : D
Destination : 2001:DB8:4::	PrefixLength : 64
NextHop : 2001:DB8:2::2	Preference : 60
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:db8:5::	PrefixLength : 64
NextHop : 2001:db8:5::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/3	Flags : D
Destination : 2001:db8:5::1	PrefixLength : 128
NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/3	Flags : D
Destination : 2001:db8:6::	PrefixLength : 64
NextHop : 2001:db8:2::2	Preference : 60
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:db8:6::	PrefixLength : 64
NextHop : 2001:db8:1::2	Preference : 60
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/2	Flags : D
Destination : FE80::	PrefixLength : 10
NextHop : ::	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : NULL0	Flags : D

Step 3 Change the priorities of the IPv6 static routes.

Change the preference value of the IPv6 static route on Device D.

```
<DeviceD> system-view
[~DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet1/0/2 2001:db8:1::2 preference 40
[*DeviceD] commit
[~DeviceD] quit
```

Check the routing table of Device D. The following command output shows that the preference value of the IPv6 static route has been changed.

```
<DeviceD> display ipv6 routing-table
Routing Table : _public_
Destinations : 13      Routes : 13

Destination : ::1          PrefixLength : 128
```

NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : InLoopBack0	Flags : D
Destination : ::FFFF:127.0.0.0	PrefixLength : 104
NextHop : ::FFFF:127.0.0.1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : InLoopBack0	Flags : D
Destination : ::FFFF:127.0.0.1	PrefixLength : 128
NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : InLoopBack0	Flags : D
Destination : 2001:db8:1::	PrefixLength : 64
NextHop : 2001:db8:1::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/2	Flags : D
Destination : 2001:db8:1::1	PrefixLength : 128
NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/2	Flags : D
Destination : 2001:DB8:2::	PrefixLength : 64
NextHop : 2001:DB8:2::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:DB8:2::1	PrefixLength : 128
NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:DB8:3::	PrefixLength : 64
NextHop : 2001:DB8:1::2	Preference : 60
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/2	Flags : D
Destination : 2001:DB8:4::	PrefixLength : 64
NextHop : 2001:DB8:2::2	Preference : 60
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/1	Flags : D
Destination : 2001:db8:5::	PrefixLength : 64
NextHop : 2001:db8:5::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/3	Flags : D
Destination : 2001:db8:5::1	PrefixLength : 128
NextHop : ::1	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : GigabitEthernet1/0/3	Flags : D
Destination : 2001:db8:6::	PrefixLength : 64
NextHop : 2001:db8:1::2	Preference : 40
Cost : 0	Protocol : Static
RelayNextHop : ::	TunnelID : 0x0

Interface : GigabitEthernet1/0/2	Flags : D
Destination : FE80::	PrefixLength : 10
NextHop : ::	Preference : 0
Cost : 0	Protocol : Direct
RelayNextHop : ::	TunnelID : 0x0
Interface : NULL0	Flags : D

Step 4 Enable FRR for IPv6 static routes.

Enable FRR for static route on Device D.

```
<DeviceD> system-view
[~DeviceD] ipv6 route-static frr
[*DeviceD] commit
[~DeviceD] quit
```

Check the backup outbound interface and the backup next hop on Device D.

```
<DeviceD> display ipv6 routing-table 2001:db8:6:: verbose
Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:6:: PrefixLength : 64
NextHop : 2001:db8:1::2 Preference : 40
Neighbour : :: ProcessID : 0
Label : NULL Protocol : Static
State : Active Adv Cost : 0
Entry ID : 0 EntryFlags : 0x00000000
Reference Cnt: 0 Tag : 0
Priority : medium Age : 28sec
IndirectID : 0xFC000105
RelayNextHop : :: TunnelID : 0x0
Interface : GigabitEthernet1/0/2 Flags : D
BkNextHop : 2001:db8:2::2 BklInterface : GigabitEthernet1/0/1
BkLabel : NULL BkTunnelID : 0x0
BkPETunnelID : 0x0 BkIndirectID : 0xFC0001
```

Step 5 Configure static BFD for IPv6 static routes.

- Configure a BFD session.

On Device D, configure a BFD session between Device D and Device C.

```
<DeviceD> system-view
[~DeviceD] bfd
[*DeviceD-bfd] quit
[*DeviceD] bfd aa bind peer-ipv6 2001:db8:3::2 source-ipv6 2001:db8:1::1
[*DeviceD-bfd-session-aa] discriminator local 10
[*DeviceD-bfd-session-aa] discriminator remote 20
[*DeviceD-bfd-session-aa] commit
[~DeviceD-bfd-session-aa] quit
```

On Device C, configure a BFD session between Device C and Device D.

```
<DeviceC> system-view
[~DeviceC] bfd
[*DeviceC-bfd] quit
[*DeviceC] bfd ab bind peer-ipv6 2001:db8:1::1 source-ipv6 2001:db8:3::2
[*DeviceC-bfd-session-ab] discriminator local 20
[*DeviceC-bfd-session-ab] discriminator remote 10
[*DeviceC-bfd-session-ab] commit
[~DeviceC-bfd-session-ab] quit
```

- Configure a static route and bind it to the BFD session.

On Device D, configure a static route and bind it to the BFD session named **aa**.

```
[~DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet1/0/2 2001:db8:1::2 preference 40
track bfd-session aa
```

Step 6 Simulate a fault on Link A.

```
[~DeviceD] interface GigabitEthernet1/0/2
[~DeviceD-GigabitEthernet1/0/2] shutdown
[~DeviceD-GigabitEthernet1/0/2] commit
[~DeviceD-GigabitEthernet1/0/2] quit
[~DeviceD] quit
```

Check the routes to 2001:db8:6::/64 on Device D.

```
<DeviceD> display ipv6 routing-table 2001:db8:6:: verbose
Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:6:: PrefixLength : 64
NextHop    : 2001:db8:2::2 Preference  : 60
Neighbour   : :: ProcessID   : 0
Label       : NULL Protocol    : Static
State       : Active Adv Cost        : 0
Entry ID    : 0 EntryFlags  : 0x00000000
Reference Cnt: 0 Tag          : 0
Priority    : medium Age         : 43sec
IndirectID  : 0xFC000106 TunnelID    : 0x0
RelayNextHop: :: Flags        : D
Interface   : GigabitEthernet1/0/1
```

----End

Configuration Files

- Device D configuration file

```
#
sysname DeviceD
#
bfd
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::1/64
#
bfd aa bind peer-ipv6 2001:db8:3::2 source-ipv6 2001:db8:1::1
discriminator local 10
discriminator remote 20
#
ipv6 route-static frr
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet1/0/2 2001:db8:1::2 preference 40 track bfd-session aa
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet1/0/1 2001:db8:2::2
ipv6 route-static 2001:db8:3:: 64 GigabitEthernet1/0/2 2001:db8:1::2
ipv6 route-static 2001:db8:4:: 64 GigabitEthernet1/0/1 2001:db8:2::2
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/2
undo shutdown
```

```
ipv6 enable
ipv6 address 2001:db8:1::2/64
#
interface GigabitEthernet2/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
#
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet1/0/2 2001:db8:1::1
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet2/0/3 2001:db8:3::2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
#
interface GigabitEthernet3/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
#
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet1/0/1 2001:db8:2::1
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet3/0/3 2001:db8:4::2
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
bfd
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:6::1/64
#
interface GigabitEthernet2/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::2/64
#
interface GigabitEthernet3/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::2/64
#
bfd ab bind peer-ipv6 2001:db8:1::1 source-ipv6 2001:db8:3::2
discriminator local 20
discriminator remote 10
#
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet2/0/3 2001:db8:3::1
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet3/0/3 2001:db8:4::1
ipv6 route-static 2001:db8:1:: 64 GigabitEthernet2/0/3 2001:db8:3::1
ipv6 route-static 2001:db8:2:: 64 GigabitEthernet3/0/3 2001:db8:4::1
#
return
```

1.1.4 OSPF Configuration

1.1.4.1 OSPF Description

1.1.4.1.1 Overview of OSPF

Definition

Open Shortest Path First (OSPF) is a link-state Interior Gateway Protocol (IGP) developed by the Internet Engineering Task Force (IETF).

OSPF version 2 (OSPFv2) is intended for IPv4. OSPF version 3 (OSPFv3) is intended for IPv6.

NOTE

In this document, OSPF refers to OSPFv2, unless otherwise stated.

Purpose

Before the emergence of OSPF, the Routing Information Protocol (RIP) was widely used as an IGP on networks. RIP is a distance-vector routing protocol. Due to its slow convergence, routing loops, and poor scalability, RIP is gradually being replaced with OSPF.

Typical IGPs include RIP, OSPF, and Intermediate System to Intermediate System (IS-IS). **Table 1-15** describes differences among the three typical IGPs.

Table 1-15 Differences among RIP, OSPF, and IS-IS

Item	RIP	OSPF	IS-IS
Protocol type	IP layer protocol	IP layer protocol	Link layer protocol
Application scope	Applies to small networks with simple architectures, such as campus networks.	Applies to medium-sized networks with several hundred routers supported, such as enterprise networks.	Applies to large networks, such as Internet service provider (ISP) networks.

Item	RIP	OSPF	IS-IS
Routing algorithm	Uses a distance-vector algorithm and exchanges routing information over the User Datagram Protocol (UDP).	Uses the shortest path first (SPF) algorithm to generate a shortest path tree (SPT) based on the network topology, calculates shortest paths to all destinations, and exchanges routing information over IP.	Uses the SPF algorithm to generate an SPT based on the network topology, calculates shortest paths to all destinations, and exchanges routing information over IP. The SPF algorithm runs separately in Level-1 and Level-2 databases.
Route convergence speed	Slow	Less than 1 second	Less than 1 second
Scalability	Not supported	Supported by partitioning a network into areas	Supported by defining router levels

Benefits

OSPF offers the following benefits:

- Wide application scope: OSPF applies to medium-sized networks with several hundred routers, such as enterprise networks.
- Network masks: OSPF packets can carry masks, and therefore the packet length is not limited by natural IP masks. OSPF can process variable length subnet masks (VLSMs).
- Fast convergence: When the network topology changes, OSPF immediately sends link state update (LSU) packets to synchronize the changes to the link state databases (LSDBs) of all routers in the same autonomous system (AS).
- Loop-free routing: OSPF uses the SPF algorithm to calculate loop-free routes based on the collected link status.
- Area partitioning: OSPF allows an AS to be partitioned into areas, which simplifies management. Routing information transmitted between areas is summarized, which reduces network bandwidth consumption.
- Equal-cost routes: OSPF supports multiple equal-cost routes to the same destination.
- Hierarchical routing: OSPF uses intra-area routes, inter-area routes, Type 1 external routes, and Type 2 external routes, which are listed in descending order of priority.
- Authentication: OSPF supports area-based and interface-based packet authentication, which ensures packet exchange security.

- Multicast: OSPF uses multicast addresses to send packets on certain types of links, which minimizes the impact on other devices.

1.1.4.1.2 Understanding OSPF

Basic Concepts of OSPF

Router ID

A router ID is a 32-bit unsigned integer and uniquely identifies a router in an AS. A router ID must exist on a router if OSPF needs to be run on it.

A router ID can be manually configured or automatically selected by the router.

If a router ID is specified when an OSPF process is created in the system view, the configured router ID is used. If no router ID is specified when an OSPF process is created, the OSPF process uses the router ID in RM. For details about router ID selection rules in RM, see the **router id** command in the system view. In any of the following situations, OSPF router ID re-selection is performed:

- After you run this command to configure a new OSPF router ID, the OSPF process automatically restarts.
- No router ID is specified for an OSPF process, the router ID in RM changes, and the OSPF process is restarted using the **reset ospf process** command.
- If the self-healing function is enabled for OSPF router ID conflicts and an OSPF router ID conflict occurs on the network, a new OSPF router ID is selected and the OSPF process is automatically restarted. If no router ID is specified when an OSPF process is created and the device restarts in CFG mode, the router ID in the RM module when the OSPF process is created is used as the OSPF router ID after the restart. Therefore, you are advised to manually specify a router ID when creating the OSPF process.

Areas

When a large number of routers run OSPF, link state databases (LSDBs) become very large and require a large amount of storage space. Large LSDBs also complicate shortest path first (SPF) computation and overload the routers. As the network scale expands, there is an increasing probability that the network topology changes, causing the network to change continuously. In this case, a large number of OSPF packets are transmitted on the network, leading to a decrease in bandwidth utilization efficiency. In addition, each time the topology changes, all routers on the network must recalculate routes.

OSPF prevents frequent LSDB updates and improves network utilization by partitioning an AS into different areas. routers can be logically allocated to different groups (areas), and each group is identified by an area ID. A router, not a link, resides at the border of an area. A network segment or link can belong to only one area. An area must be specified for each OSPF interface.

OSPF areas include common areas, stub areas, and not-so-stubby areas (NSSAs). **Table 1-16** describes these OSPF areas.

Table 1-16 Area types

Area Type	Function	Description
Common area	<p>By default, OSPF areas are defined as common areas. Common areas include:</p> <ul style="list-style-type: none"> Standard area is the most prevalent area and transmits intra-area, inter-area, and external routes. Backbone area connects to all other OSPF areas and is usually represented by Area 0. The backbone area is responsible for inter-area routing. Routing information between non-backbone areas must be forwarded through the backbone area. 	<ul style="list-style-type: none"> The backbone area must have all its devices connected. All non-backbone areas must remain connected to the backbone area.
Stub area	<p>A stub area is a non-backbone area with only one area border router (ABR) and generally resides at the border of an AS. The ABR in a stub area does not transmit received AS external routes, which significantly decreases the number of entries in the routing table on the router and the amount of routing information to be transmitted. To ensure the reachability of AS external routes, the ABR in the stub area generates a default route and advertises the route to non-ABRs in the stub area.</p> <p>A totally stub area allows only intra-area routes and ABR-advertised Type 3 default routes to be advertised within the area and does not allow AS external routes or inter-area routes to be advertised.</p>	<ul style="list-style-type: none"> The backbone area cannot be configured as a stub area. An autonomous system boundary router (ASBR) cannot exist in a stub area. Therefore, AS external routes cannot be advertised within the stub area. A virtual link cannot pass through a stub area.
NSSA	<p>An NSSA is similar to a stub area. An NSSA does not support Type 5 LSAs but can import AS-external routes. Type 7 LSAs carrying information about the AS-external routes are generated by ASBRs in the NSSA and are advertised only within the NSSA. When the Type 7 LSAs reach an ABR in the NSSA, the ABR translates the Type 7 LSAs into Type 5 LSAs and floods them to the entire OSPF domain.</p> <p>A totally NSSA allows only intra-area routes to be advertised within the area. AS external routes or inter-area routes cannot be advertised in a totally NSSA.</p>	<ul style="list-style-type: none"> An ASBR in an NSSA advertises Type 7 LSA default routes within the NSSA. All inter-area routes must be advertised by ABRs. A virtual link cannot pass through an NSSA.

Router Types

Routers are classified by location in an AS. [Figure 1-59](#) and [Table 1-17](#) show the classification.

Figure 1-59 Router types

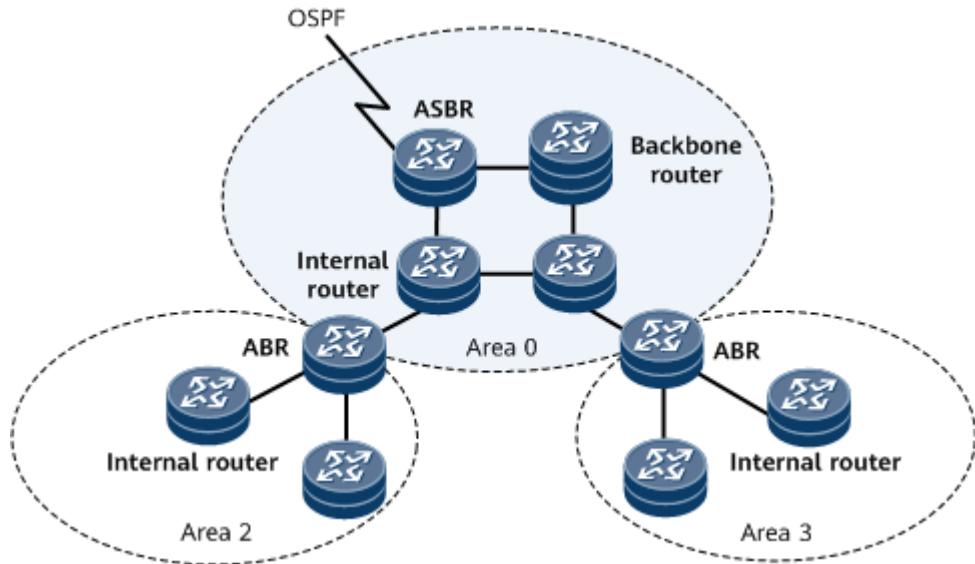


Table 1-17 Router types

Router Type	Description
Internal router	All interfaces of an internal router belong to the same OSPF area.
ABR	An ABR can belong to two or more areas, one of which must be a backbone area. An ABR connects the backbone area and non-backbone areas, and it can connect to the backbone area either physically or logically.
Backbone router	At least one interface on this type of router belongs to the backbone area. Internal routers in the backbone area and all ABRs are backbone routers.
ASBR	Exchanges routing information with other ASs. An ASBR may be an internal router or an ABR, and therefore may not necessarily reside at the border of an AS.

LSA

OSPF encapsulates routing information into LSAs for transmission. [Table 1-18](#) describes LSAs and their functions.

Table 1-18 Different types of LSAs and their functions

LSA Type	LSA Function
Router-LSA (Type 1)	Describes the link status and cost of a router. Router-LSAs are generated by each router and advertised within the area to which the router belongs.
Network-LSA (Type 2)	Describes the link status on the local network segment. Network-LSAs are generated by a designated router (DR) and advertised within the area to which the DR belongs.
Network-Summary-LSA (Type 3)	Describes routes on a network segment of an area. Network-Summary-LSAs are generated by an ABR and advertised to other areas, excluding totally stub areas and totally NSSAs. For example, an ABR belongs to both area 0 and area 1, area 0 has a network segment 10.1.1.0, and area 1 has a network segment 10.2.1.0. In this case, the ABR generates Type 3 LSAs destined for the network segment 10.2.1.0 for area 0, and Type 3 LSAs destined for the network segment 10.1.1.0 for area 1.
ASBR-Summary-LSA (Type 4)	Describes routes of an area to the ASBRs of other areas. ASBR-Summary-LSAs are generated by an ABR and advertised to other areas, excluding stub areas, totally stub area, NSSAs, and totally NSSAs.
AS-external-LSA (Type 5)	Describes AS external routes. AS-external-LSAs are generated by an ASBR and are advertised to all areas, excluding stub areas, totally stub areas, NSSAs, and totally NSSAs.
NSSA LSA (Type 7)	Describes AS external routes. NSSA-LSAs are generated by an ASBR and advertised only within NSSAs.
Opaque-LSA (Type 9/Type 10/Type 11)	<p>Opaque-LSAs provide a general mechanism for OSPF extension.</p> <ul style="list-style-type: none"> • Type 9 LSAs are advertised only on the network segment where the interface advertising the LSAs resides. The Grace LSAs used in graceful restart (GR) are one type of Type 9 LSA. • Type 10 LSAs are advertised within an OSPF area. The LSAs that are used to support traffic engineering (TE) are one type of Type 10 LSA. • Type 11 LSAs are advertised in an AS. The LSAs used to support routing loop detection for routes imported to OSPF are one type of Type 11 LSA.

Table 1-19 describes whether a type of LSA is supported in an area.

Table 1-19 Support status of LSAs in different types of areas

Area Type	Router-LSA (Type 1)	Network-LSA (Type 2)	Network-Summary-LSA (Type 3)	ASBR-Summary-LSA (Type 4)	AS-external-LSA (Type 5)	NSSA-LSA (Type 7)
Common area (including standard and backbone areas)	Supported	Supported	Supported	Supported	Supported	Not supported
Stub area	Supported	Supported	Supported	Not supported	Not supported	Not supported
Totally stub area	Supported	Supported	Not supported (except the default Type 3 LSA)	Not supported	Not supported	Not supported
NSSA	Supported	Supported	Supported	Not supported	Not supported	Supported
Totally NSSA	Supported	Supported	Not supported (except the default Type 3 LSA)	Not supported	Not supported	Supported

Packet Types

OSPF uses IP packets to encapsulate protocol packets. The protocol number is 89. OSPF packets are classified as Hello, database description (DD), link state request (LSR), link state update (LSU), or link state acknowledgment (LSAck) packets, as described in [Table 1-20](#).

Table 1-20 OSPF packets and their functions

Packet Type	Function
Hello packet	Hello packets are sent periodically to discover and maintain OSPF neighbor relationships.
Database description (DD) packet	A DD packet contains the summaries of LSAs in the local LSDB. DD packets are used for LSDB synchronization between two routers.
Link state request (LSR) packet	LSR packets are sent to OSPF neighbors to request required LSAs. A router sends LSR packets to its OSPF neighbor only after DD packets have been successfully exchanged.
Link state update (LSU) packet	LSU packets are used to transmit required LSAs to OSPF neighbors.
Link state acknowledgment (LSAck) packet	LSAck packets are used to acknowledge received LSAs.

Route Types

AS intra-area and inter-area routes describe the network structure within an AS, and AS external routes describe how to select routes to destinations outside the AS. OSPF classifies the imported AS external routes into Type 1 and Type 2.

Table 1-21 describes OSPF routes in descending order of priority.

Table 1-21 Route types

Route Type	Description
Intra Area	Intra-area route
Inter Area	Inter-area routes
Type 1 external route	This type of route is more reliable. Cost of a Type 1 external route = Cost of the route from a router to an ASBR + Cost of the route from the ASBR to the destination When multiple ASBRs exist, the cost of each Type 1 external route equals the cost of the route from the local device to an ASBR plus the cost of the route from the ASBR to the destination. The cost is used for route selection.

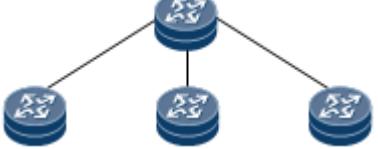
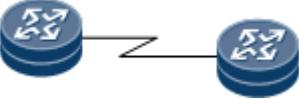
Route Type	Description
Type 2 external route	<p>Because a Type 2 external route offers low reliability, its cost is considered to be much greater than the cost of any internal route to an ASBR.</p> <p>Cost of a Type 2 external route = Cost of the route from an ASBR to the destination</p> <p>If multiple ASBRs have routes to the same destination, the route with the lowest cost from the corresponding ASBR to the destination is selected and imported. If the costs of the imported routes are the same, the router compares the costs of the routes from the local router to the corresponding ASBR and selects the route with the smallest cost to import. The cost of each Type 2 external route equals the cost of the route from the corresponding ASBR to the destination.</p>

OSPF Network Classification

According to the link layer protocol type, OSPF classifies networks into four types, as described in [Table 1-22](#).

Table 1-22 OSPF network classification

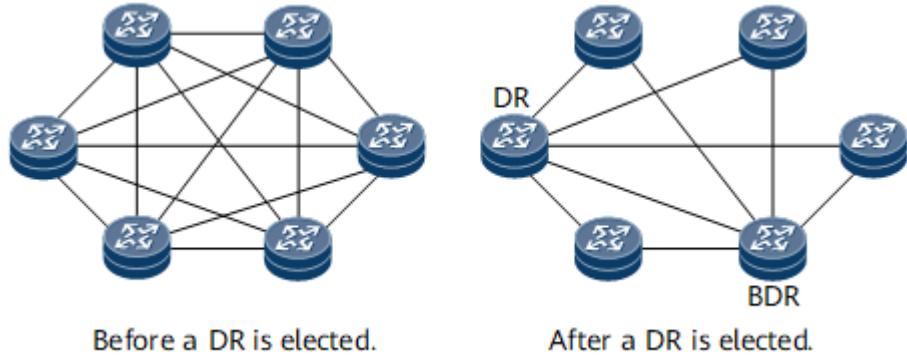
Network Type	Link Layer Protocol	Graph
Broadcast	<ul style="list-style-type: none"> • Ethernet • FDDI 	
NBMA	X.25	

Network Type	Link Layer Protocol	Graph
Point-to-Multipoint (P2MP)	No link layer protocol is considered as the P2MP type by default. P2MP is forcibly changed from another type of network. In most cases, a non-fully meshed NBMA network is changed to a P2MP network.	
P2P	<ul style="list-style-type: none"> • PPP • LAPB 	

DR and BDR

On broadcast or NBMA networks, any two routers need to exchange routing information. As shown in [Figure 1-60](#), n routers are deployed on the network. $n \times (n - 1)/2$ adjacencies must be established. Any route change on a router is transmitted to other routers, which wastes bandwidth resources. OSPF resolves this problem by defining a DR and a BDR. After a DR is elected, all routers send routing information only to the DR. Then the DR broadcasts LSAs. routers other than the DR and BDR are called DR others. The DR others establish only adjacencies with the DR and BDR and not with each other. This process reduces the number of adjacencies established between routers on broadcast or NBMA networks.

Figure 1-60 Network topologies before and after a DR election



If the original DR fails, routers must reelect a DR and the routers except the new DR must synchronize routing information to the new DR. This process is lengthy, which may cause incorrect route calculations. A BDR is used to shorten the process. The BDR is a backup for a DR. A BDR is elected together with a DR. The BDR establishes adjacencies with all routers on the network segment and exchanges routing information with them. If the DR fails, the BDR immediately becomes a new DR. Because no re-election is required and adjacencies have been

established, this process is very short. In this case, a new BDR needs to be elected. Although this process takes a long time, it does not affect route calculation.

The DR and BDR are not designated manually. Instead, they are elected by all routers on the network segment. The DR priority of an interface on the router determines whether the interface is qualified for DR or BDR election. On the local network segment, the routers whose DR priorities are greater than 0 are all candidates. Hello packets are used for the election. Each router adds information about the DR elected by itself to a Hello packet and sends the packet to other routers on the network segment. If two routers on the same network segment declare that they are DRs, the one with a higher DR priority wins. If they have the same priority, the one with a larger router ID wins. If the priority of a router is 0, it cannot be elected as a DR or BDR.

OSPF Multi-Process

OSPF multi-process allows multiple OSPF processes to independently run on the same router. Route exchange between different OSPF processes is similar to that between different routing protocols. A router's interface can belong only to one OSPF process.

A typical application of OSPF multi-process is that OSPF runs between PEs and CEs in VPN scenarios and OSPF is also used as an IGP on the VPN backbone network. The OSPF processes on the PEs are independent of each other.

OSPF Default Route

A default route is the route whose destination address and mask are both all 0s. If no matching route is found, the default route can be used by the router to forward packets.

OSPF default routes are generally applied to the following scenarios:

- An ABR in an area advertises Type 3 default summary LSAs within the area to help the routers in the area forward inter-area packets.
- An ASBR in an AS advertises Type 5 external default ASE LSAs or Type 7 external default NSSA LSAs to help the routers in the AS forward AS external packets.

OSPF routes are hierarchically managed. The priority of the default route carried in Type 3 LSAs is higher than the priority of the default route carried in Type 5 or Type 7 LSAs.

The rules for advertising OSPF default routes are as follows:

- An OSPF device can advertise default route LSAs only when it has an external interface.
- If an OSPF device has advertised default route LSAs, it no longer learns the same type of default route advertised by other routers. That is, the device no longer calculates the same type of default route LSA advertised by other routers. However, the corresponding LSAs exist in the database.
- If the advertisement of external default routes depends on other routes, the dependent routes cannot be the routes (learned by the local OSPF process) in the local OSPF routing domain. This is because external default routes are used to guide packet forwarding outside the domain. However, the next hops

of routes in the OSPF routing domain are within the domain, unable to guide packet forwarding outside the domain.

- Before a router advertises a default route, it checks whether a neighbor in the full state is present in area 0. The router advertises a default route only when a neighbor in the full state is present in area 0. If no such a neighbor exists, the backbone area cannot forward packets and advertising a default route is meaningless. For the concept of the Full State, see [OSPF Neighbor States](#).

Table 1-23 describes the principles for advertising default routes in different areas.

Table 1-23 Principles for advertising default routes in different areas

Area Type	Principles for Advertising Default Routes
Common area	<p>By default, OSPF devices in a common area do not generate default routes, even if they have default routes.</p> <p>When a default route is generated by another routing process, the router must advertise the default route to the entire OSPF AS. To achieve this, a command must be run on the ASBR to generate a default route. After the configuration is complete, the router generates a default ASE LSA (Type 5 LSA) and advertises it to the entire OSPF AS.</p> <p>If no default route exists on the ASBR, the router does not advertise a default route.</p>
Stub Area	<p>Type 5 LSAs cannot be advertised within a stub area.</p> <p>A router in the stub area must learn AS external routes from an ABR. The ABR automatically generates a default Summary LSA (Type 3 LSA) and advertises it within the entire stub area. Then the device can learn AS external routes from the ABR.</p>
Totally Stub Area	<p>Neither Type 3 (except default Type 3 LSAs) nor Type 5 LSAs can be advertised within a totally stub area.</p> <p>A router in the totally stub area must learn AS external and inter-area routes from an ABR. After you configure a totally stub area, an ABR automatically generates a default Summary LSA (Type 3 LSA) and advertises it within the entire totally stub area. Then the device can learn AS external and inter-area routes from the ABR.</p>

Area Type	Principles for Advertising Default Routes
NSSA	<p>A small number of AS external routes learned from the ASBR in an NSSA can be imported to the NSSA. External routes ASE LSAs (Type 5 LSAs) to other areas cannot be advertised within the NSSA. When at least a neighbor in Full status and an interface that is Up exist in the backbone area, the ABR automatically generates a Type 7 LSA carrying a default route and advertises it within the entire NSSA. In this case, a small number of routes are learned through the ASBR in the NSSA, and other routes are learned through the ABR in the NSSA. You can manually configure the ASBR to generate a default NSSA LSA (Type 7 LSA) and advertise it in the entire NSSA. In this manner, external routes can also be learned through the ASBR in the NSSA.</p> <p>An ABR does not translate Type 7 LSA default routes into Type 5 LSA default routes for transmission in the entire OSPF domain.</p>
Totally NSSA	<p>A totally NSSA does not allow ASE LSAs (Type 5 LSAs) of external routes or inter-area routes (Type 3 LSAs, except the default Type 3 LSAs) to be transmitted within the area.</p> <p>A router in this area must learn AS external routes from an ABR. After a totally NSSA is configured, the ABR automatically generates default Type 3 LSAs and Type 7 LSAs and advertises them within the entire totally NSSA. Then, AS external and inter-area routes can be advertised within the area through the ABR.</p>

OSPF Fundamentals

OSPF route calculation involves the following processes:

1. **Adjacency establishment**

The adjacency establishment process is as follows:

- a. The local and remote devices use OSPF interfaces to exchange Hello packets to establish a Neighbor relationship.
- b. The local and remote devices negotiate a master/slave relationship and exchange Database Description (DD) packets.
- c. The local and remote devices exchange link state advertisements (LSAs) to synchronize their link state databases (LSDBs).

2. **Route calculation:** OSPF uses the shortest path first (SPF) algorithm to calculate routes, implementing fast route convergence.

OSPF Neighbor States

To exchange routing information on an OSPF network, devices must establish adjacencies. The differences between neighbor relationships and adjacencies are described as follows:

- Neighbor relationship: After the local router starts, it uses an OSPF interface to send a Hello packet to the remote router. After the remote router receives

the packet, it checks whether the parameters carried in the packet are consistent with its own parameters. If the parameters carried in the packet are consistent with its own parameters, the remote router establishes a neighbor relationship with the local router.

- Adjacency: After two devices establish a neighbor relationship, they exchange DD packets and LSAs to establish an adjacency.

OSPF has eight neighbor states: Down, Attempt, Init, 2-way, Exstart, Exchange, Loading, and Full. Down, 2-way, and Full are stable states. Attempt, Init, Exstart, Exchange, and Loading are unstable states, which last only several minutes. [Figure 1-61](#) shows the eight neighbor states.

Figure 1-61 OSPF neighbor states

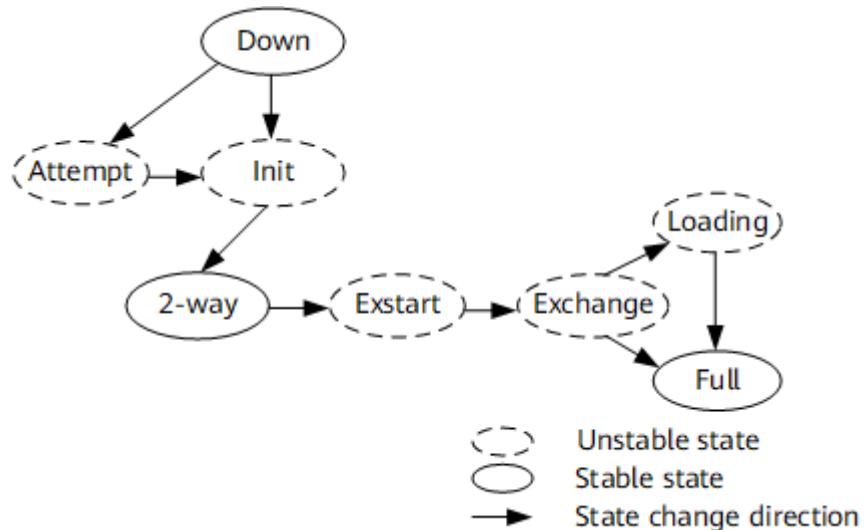


Table 1-24 OSPF neighbor states and their meanings

OSPF Neighbor State	Description
Down	This is the initial state of a neighbor conversation. This state indicates that no Hello packet is received from the neighbor within the dead interval.
Attempt	In this state, the device periodically sends Hello packets to manually configured neighbors. NOTE This state applies only to non-broadcast multiple access (NBMA) interfaces.
Init	This state indicates that the local end has received Hello packets from its neighbor, but the neighbor does not receive Hello packets from the local end.

OSPF Neighbor State	Description
2-way	This state indicates that a device has received Hello packets from its neighbors and Neighbor relationship have been established between the devices. If no adjacency needs to be established, the neighbors remain in the 2-way state. If adjacencies need to be established, the neighbors enter the Exstart state.
Exstart	In the Exstart state, devices establish a master/slave relationship to ensure that DD packets are sequentially exchanged.
Exchange	In the Exchange state, routers exchange DD packets. A router uses a DD packet to describe its own LSDB and sends the packet to its neighbors.
Loading	In the Loading state, a device sends Link State Request (LSR) packets to its neighbors to request their LSAs for LSDB synchronization.
Full	In this state, a device establishes adjacencies with its OSPF neighbors and all LSDBs have been synchronized.

NOTE

The neighbor state of the local device may be different from that of a remote device. For example, the neighbor state of the local router is Full, but the neighbor state of the remote router is Loading.

Adjacency Establishment

Adjacencies can be established in either of the following situations:

- Two routers have established a neighbor relationship and communicate for the first time.
- The designated router (DR) or backup designated router (BDR) on a network segment changes.

The OSPF adjacency establishment process varies according to the network type (broadcast, NBMA, P2P, or P2MP).

OSPF adjacency establishment on a broadcast network

[Figure 1-62](#) shows the adjacency establishment process on a broadcast network.

On the broadcast network, the DR and BDR establish adjacencies with each router on the same network segment, but DR others establish only neighbor relationships.

Figure 1-62 OSPF adjacency establishment process on a broadcast network

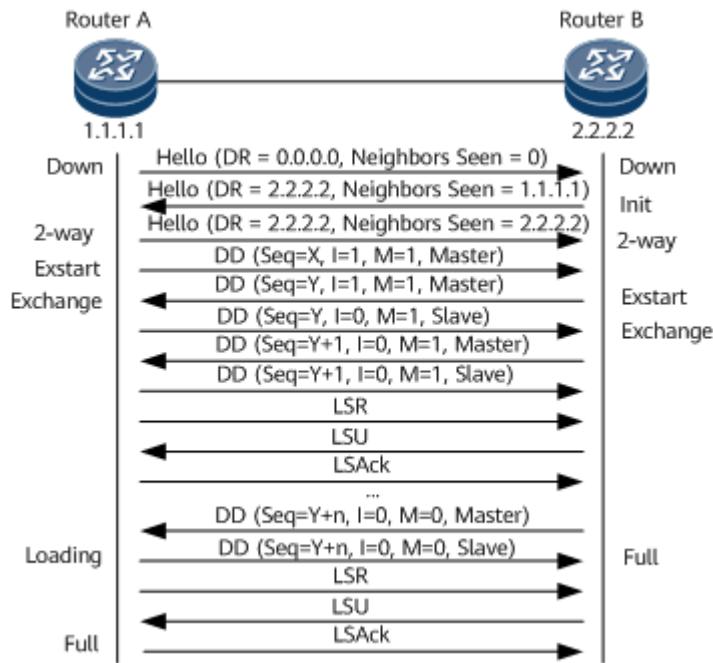


Figure 1-62 shows the OSPF adjacency establishment process on a broadcast network.

1. Neighbor relationship establishment

- Router A uses the multicast address 224.0.0.5 to send a Hello packet through the OSPF interface connected to a broadcast network. In this case, Router A does not know which router is the DR or which router is a neighbor. Therefore, the DR field is 0.0.0.0, and the Neighbors Seen field is 0.
- After Router B receives the packet, it returns a Hello packet to Router A. The returned packet carries the DR field of 2.2.2.2 (ID of Router B) and the Neighbors Seen field of 1.1.1.1 (Router A's router ID). Router A has been discovered but its router ID is less than that of Router B, and therefore Router B regards itself as a DR. Then Router B's state changes to Init.
- After Router A receives the packet, Router A's state changes to 2-way.

NOTE

The following procedures are not performed for DR others on a broadcast network.

2. Master/Slave negotiation and DD packet exchange

- Router A sends a DD packet. The packet carries the following fields:
 - Seq field: The value x indicates the sequence number is x .
 - I field: The value 1 indicates that the packet is the first DD packet, which is used to negotiate a master/slave relationship and does not carry LSA summaries.
 - M field: The value 1 indicates that the packet is not the last DD packet.

- MS field: The value 1 indicates that Router A declares itself a master.

To improve transmission efficiency, Router A and Router B determine which LSAs in each other's LSDB need to be updated. If one party determines that an LSA of the other party is already in its own LSDB, it does not send an LSR packet for updating the LSA to the other party. To achieve the preceding purpose, Router A and Router B first send DD packets, which carry summaries of LSAs in their own LSDBs. Each summary identifies an LSA. To ensure packet transmission reliability, a master/slave relationship must be determined during DD packet exchange. One party serving as a master uses the Seq field to define a sequence number. The master increases the sequence number by one each time it sends a DD packet. When the other party serving as a slave sends a DD packet, it adds the sequence number carried in the last DD packet received from the master to the Seq field of the packet.

- b. After receiving the DD packet from Router A, Router B sets the neighbor state of Router A to Exstart and returns a DD packet. The returned packet does not carry LSA summaries either. Because Router B's router ID is greater than Router A's router ID, Router B declares itself a master and sets the Seq field to y .
- c. After Router A receives the DD packet, it agrees that Router B is a master and Router A's state changes to Exchange. Then Router A sends a DD packet to Router B to transmit LSA summaries. The packet carries the Seq field of y and the MS field of 0. The value 0 indicates that Router A declares itself a slave.
- d. After Router B receives the packet, Router B's state changes to Exchange and Router B sends a new DD packet containing its own LSA summaries to Router A. The value of the Seq field carried in the new DD packet is changed to $y+1$.

Router A uses the same sequence number as Router B to confirm that it has received DD packets from Router B. Router B uses the sequence number plus one to confirm that it has received DD packets from Router A. When Router B sends the last DD packet, it sets the M field of the packet to 0.

3. LSDB synchronization (through LSA requests, transmission, and response)

- a. After Router A receives the last DD packet, it finds that many LSAs in Router B's LSDB do not exist in its own LSDB, so Router A's state changes to Loading. After Router B receives the last DD packet from Router A, Router B's state directly changes to Full, because Router B's LSDB already contains all LSAs of Router A.
- b. Router A sends an LSR packet for updating LSAs to Router B. Router B returns an LSU packet to Router A. After Router A receives the packet, it sends an LSAck packet for acknowledgment.

The preceding procedures continue until the LSAs in Router A's LSDB are the same as those in Router B's LSDB. Router A sets the state of the neighbor relationship with Router B to Full. After the routers exchange DD packets and update all LSAs, the adjacency is established.

OSPF adjacency establishment on an NBMA network

The adjacency establishment process on an NBMA network is different from that on a broadcast network only before DD packets are exchanged, as marked in blue in [Figure 1-63](#).

On an NBMA network, all routers establish adjacencies only with the DR and BDR.

Figure 1-63 OSPF adjacency establishment process on an NBMA network

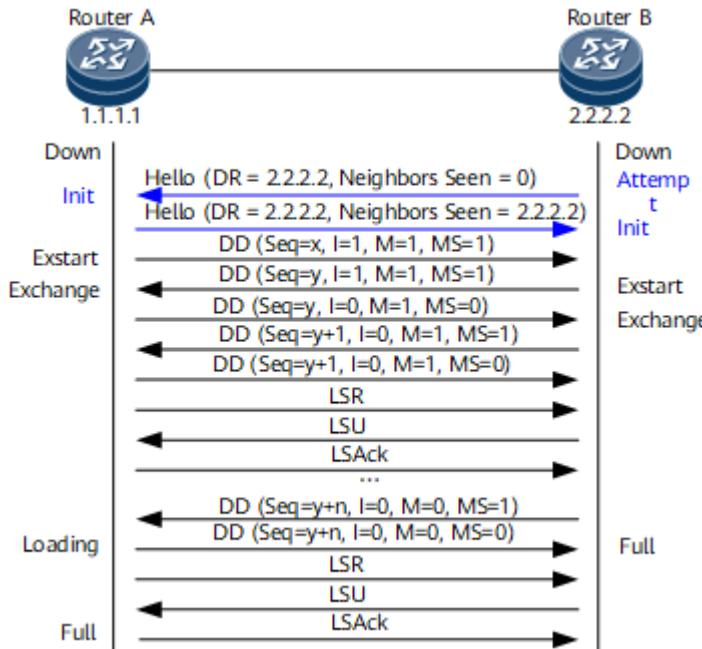


Figure 1-63 shows the process of OSPF adjacency establishment on an NBMA network.

1. Neighbor relationship establishment

- After Router B sends a Hello packet to a down interface of Router A, Router B's state changes to Attempt. At this time, Router B regards itself as a DR but does not know which router is the neighbor. The packet carries the DR field of 2.2.2.2 (ID of Router B) and the Neighbors Seen field of 0.
- After Router A receives the Hello packet, Router A sets the neighbor state to Init and returns a Hello packet. Router A agrees that Router B is a DR. The DR and Neighbors Seen fields are both set to 2.2.2.2.

NOTE

The following procedures are not performed for DR others (routers) on an NBMA network.

2. Master/Slave relationship negotiation and DD packet exchange

The procedures for negotiating a master/slave relationship and exchanging DD packets on an NBMA network are the same as those on a broadcast network.

3. The procedure for synchronizing LSDBs (through LSA requests, transmission, and response) on this type of network is the same as that on a broadcast network.

OSPF adjacency establishment on a point-to-point (P2P)/Point-to-multipoint (P2MP) network

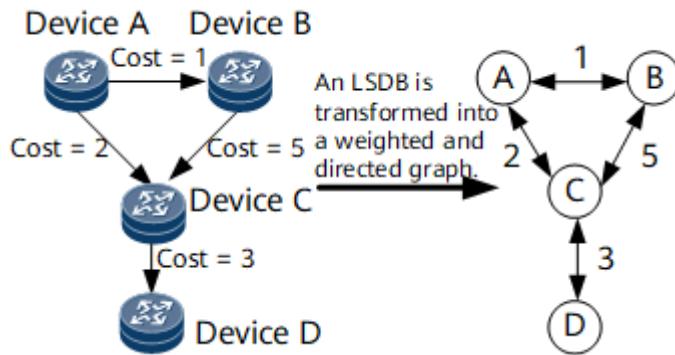
Adjacency establishment on a P2P or P2MP network is the same as that on a broadcast network except that no DR or BDR needs to be elected on the P2P or P2MP network. On a P2P network, DD packets are sent in multicast mode. On a P2MP network, DD packets are sent in unicast mode.

Route Calculation

OSPF uses the shortest path first (SPF) algorithm to calculate routes, implementing fast route convergence.

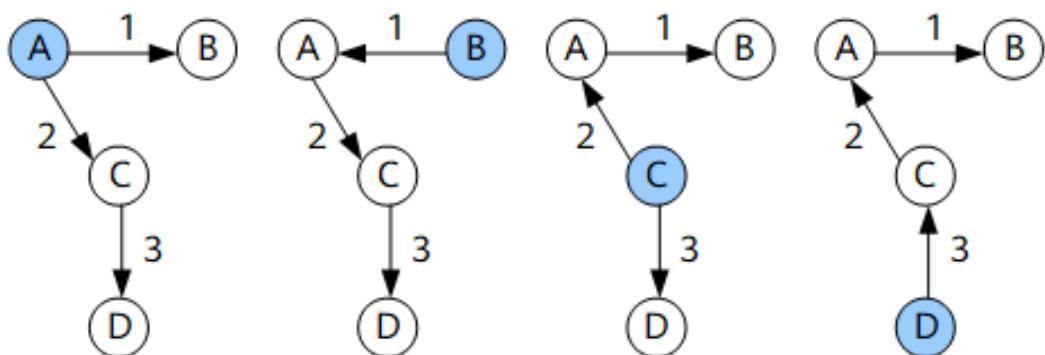
OSPF uses an LSA to describe the network topology. A Router LSA describes the attributes of a link between routers. A router transforms its LSDB into a weighted, directed graph, which reflects the topology of the entire network. All routers have the same graph. [Figure 1-64](#) shows a weighted, directed graph.

[Figure 1-64](#) Weighted, directed graph generated based on the LSDB



Based on the graph, each router uses the SPF algorithm to calculate an SPT with itself as the root. The SPT shows routes to nodes in the AS, as shown in [Figure 1-65](#).

[Figure 1-65](#) SPTs



When a router's LSDB changes, the router recalculates a shortest path. Frequent SPF calculations consume a large number of resources and affect router efficiency. Changing the interval between SPF calculations can prevent resource consumption caused by frequent LSDB changes. The default interval between SPF calculations is 5 seconds.

The route calculation process is as follows:

1. A device calculates intra-area routes.

The router uses an SPF algorithm to calculate shortest paths to other routers. Router LSAs and Network LSAs accurately describe the network topology in an area. Based on the network topology described by a Router LSA, the router calculates paths to each network segment.

 **NOTE**

If multiple equal-cost routes are produced during route calculation, the SPF algorithm retains all these routes in the LSDB.

2. The device calculates inter-area routes.

For the devices in an area, the network segment of the routes in an adjacent area is directly connected to the area border router (ABR). Because the shortest path to the ABR has been calculated in the preceding step, the devices can directly check a Network Summary LSA to obtain the shortest path to the network segment. The autonomous system boundary router (ASBR) can also be considered connected to the ABR. Therefore, the shortest path to the ASBR can also be calculated in this phase.

 **NOTE**

If the router performing an SPF calculation is an ABR, it needs to check only Network Summary LSAs in the backbone area.

If there are multiple paths to an ASBR, check whether the rules for selecting a path to the ASBR among intra-area and inter-area paths on different types of devices are the same. If the rules are different, routing loops may occur.

The RFC 1583 compatibility mode and RFC 1583 non-compatibility mode may affect path selection rules. Even in the same mode, the path selection rules on devices from different vendors may be slightly different. In this case, the rules used in RFC 1583 compatibility mode or RFC 1583 non-compatibility mode for selecting a path to an ASBR can be adjusted, preventing loops to some extent.

3. The router calculates AS external routes.

AS external routes can be considered to be directly connected to the ASBR. Because the shortest path to the ASBR has been calculated in the preceding phase, the device can check AS External LSAs to obtain the shortest paths to other ASs.

OSPF Route Control

You can use the following features to control the advertising and receiving of OSPF routes. These features meet requirements for network planning and traffic management.

- **Route summarization**

Route summarization enables a device to summarize routes with the same prefix into a single summary route and to advertise only the summary route to other areas. Route summarization reduces the size of a routing table and improves device performance.

- **Route filtering**

OSPF can use routing policies to filter routes. By default, OSPF does not filter routes.

- **OSPF Database Overflow**

Setting the maximum number of external routes in the LSDB dynamically limits the size of the OSPF LSDB and prevents the problem caused by the overflow of the OSPF LSDB.

Route Summarization

When a large OSPF network is deployed, an OSPF routing table includes a large number of routing entries. To accelerate route lookup and simplify management, configure route summarization to reduce the size of the OSPF routing table. If a link frequently alternates between Up and Down, the links not involved in the route summarization are not affected. This process prevents route flapping and improves network stability.

OSPF supports two route summarization modes.

- ABR summarization

When an ABR transmits routing information to other areas, it generates Type 3 LSAs for each network segment. If consecutive network segments exist in this area, you can summarize these network segments into a single network segment. The ABR generates one LSA for the summarized network segment and advertises only that LSA.

- ASBR summarization

If route summarization has been configured and the local router is an ASBR, the local router summarizes imported Type 5 LSAs within the summarized address range. If an NSSA is configured, imported Type 7 LSAs within the summarized address range also need to be summarized.

If the local router is both an ASBR and an ABR, it summarizes Type 5 LSAs translated from Type 7 LSAs.

Route Filtering

OSPF routing policies include access control lists (ACLs), IP prefix lists, and route-policies. For details about these policies, see the section "Routing Policy" in the *NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X Feature Description - IP Routing*.

OSPF route filtering applies in the following aspects:

- Route import

OSPF can import the routes learned by other routing protocols. You can configure a routing policy to filter routes and import only the routes that meet the filtering conditions. Only an ASBR can import routes, and therefore a routing policy for importing routes must be configured on the ASBR.

- Advertising of imported routes

OSPF advertises imported routes to its neighbors. Only an ASBR can import routes, and therefore a routing policy for the advertising of imported routes must be configured on the ASBR.

If OSPF imports a large number of external routes and advertises them to a device with a smaller routing table capacity, the device may restart unexpectedly. To prevent this problem, configure a limit on the number of imported external routes to be advertised by OSPF to ensure stable device running.

- Route learning

By configuring filtering rules, you can configure OSPF to filter received intra-area, inter-area, and AS external routes. Such filtering only determines whether to add routing entries. That is, all routes in the OSPF routing table can be calculated and advertised normally, but only the routes that match the filtering rules can be added to the local routing table.

During route learning, LSAs are not filtered. Instead, only the routes calculated based on LSAs are filtered to determine whether they are added to the routing table. Therefore, the learned LSAs are complete.

- Inter-area LSA learning

An ABR in an area can be configured to filter Summary LSAs advertised to the area. The ABR can advertise only Summary LSAs, and therefore inter-area LSA learning is valid only on the ABR.

During inter-area LSA learning, the LSAs advertised to the area are directly filtered.

- Inter-area LSA advertising

An ABR in an area can be configured to filter Summary LSAs advertised out of the local area. The ABR can advertise only Summary LSAs, and therefore inter-area LSA advertisement is valid only on the ABR.

OSPF Database Overflow

OSPF requires that devices in the same area have the same LSDB. As the number of routes increase continually, some devices cannot carry excess routing information due to limited system resources. This situation is called an OSPF database overflow.

You can configure stub areas or NSSAs to prevent resource exhaustion caused by continually increasing routing information. However, configuring stub areas or NSSAs cannot prevent an OSPF database overflow caused by a sharp increase in dynamic routes. To resolve this issue, set the maximum number of external routes supported by the LSDB to dynamically limit the LSDB's size.

 NOTE

The maximum number of external routes configured for all devices in the OSPF AS must be the same.

When the number of external routes in the LSDB reaches the maximum number, the device enters the overload state and starts the overflow timer at the same time. The device automatically exits from the overflow state after the overflow timer expires. **Table 1-25** describes the operations performed by the device after it enters or exits from the overload state.

Table 1-25 Description of the OSPF database overflow state

Phase	OSPF Processing Procedure
Staying at overload state	<p>Deletes self-generated non-default external routes and stops advertising non-default external routes.</p> <p>Discards newly received non-default external routes and does not reply with a Link State Acknowledgment (LSAck) packet.</p> <p>Checks whether the number of external routes is still greater than the configured maximum number when the overflow timer expires.</p> <ul style="list-style-type: none"> • Restarts the timer if the number of external routes is greater than or equal to the configured maximum number. • Exits from the overflow state if the number of external routes is less than the configured maximum number.
Exiting from the overflow state	<p>Ends the overflow timer.</p> <p>Advertises non-default external routes.</p> <p>Accepts newly received non-default external routes and replies with LSack packets.</p>

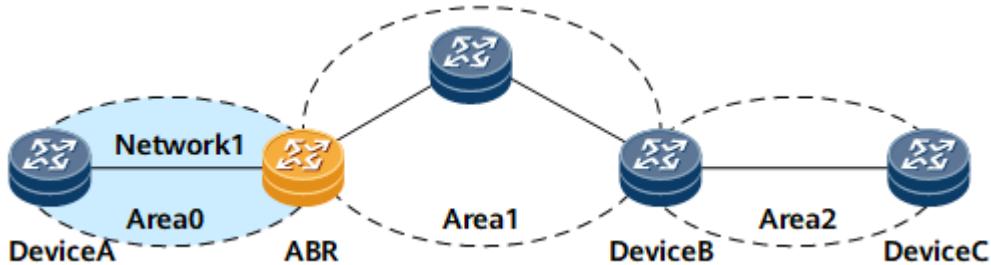
OSPF Virtual Link

Background

All non-backbone areas must be connected to the backbone area during OSPF deployment to ensure that all areas are reachable.

In [Figure 1-66](#), area 2 is not connected to area 0 (backbone area), and Device B is not an ABR. Therefore, Device B does not generate routing information about network 1 in area 0, and Device C does not have a route to network 1.

Figure 1-66 Non-backbone area not connected to the backbone area



Some non-backbone areas may not be connected to the backbone area. You can configure an OSPF virtual link to resolve this issue.

Related Concepts

A virtual link refers to a logical channel established between two ABRs over a non-backbone area.

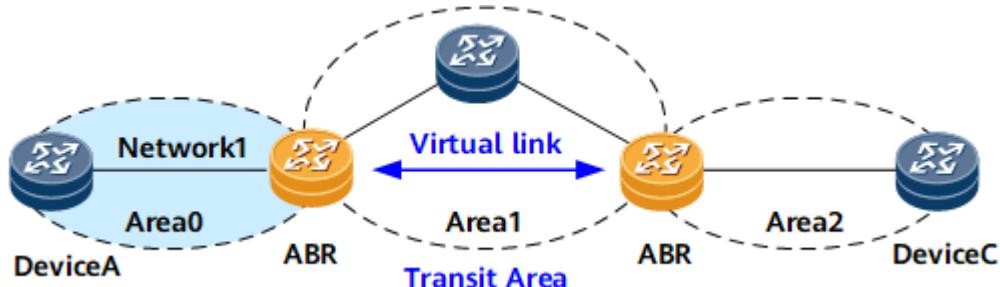
- A virtual link must be configured at both ends of the link.
- The non-backbone area involved is called a transit area.

A virtual link is similar to a point-to-point (P2P) connection established between two ABRs. You can configure interface parameters, such as the interval at which Hello packets are sent, at both ends of the virtual link as you do on physical interfaces.

Principles

In [Figure 1-67](#), two ABRs use a virtual link to directly transmit OSPF packets. The device between the two ABRs only forwards packets. Because the destination of OSPF packets is not the device, the device transparently transmits the OSPF packets as common IP packets.

Figure 1-67 OSPF virtual link

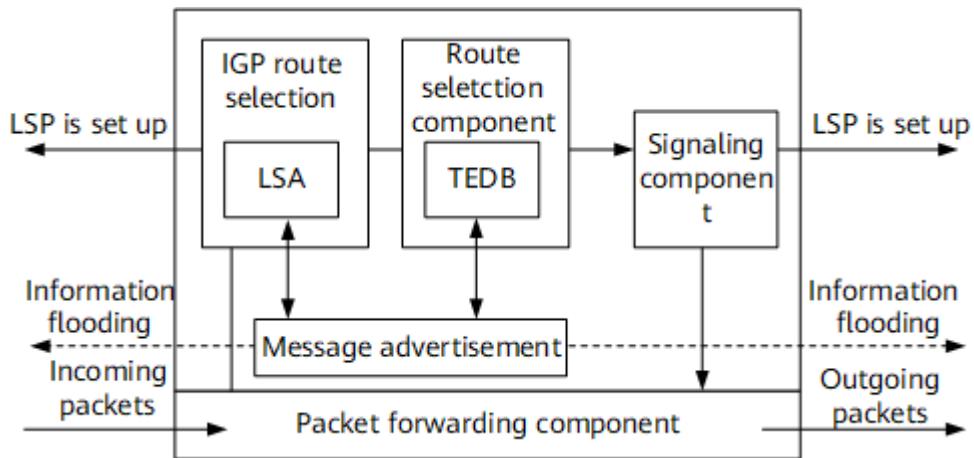


OSPF TE

OSPF Traffic Engineering (TE) is developed based on OSPF to support Multiprotocol Label Switching (MPLS) TE and establish and maintain TE LSPs. In the MPLS TE architecture described in "MPLS Feature Description", OSPF functions as the information advertising component, responsible for collecting and advertising MPLS TE information.

In addition to the network topology, TE needs to know network constraints, such as the bandwidth, TE metric, administrative group, and affinity attribute. However, current OSPF functions cannot meet these requirements. Therefore, OSPF introduces a new type of LSAs to advertise network constraints. Based on the network constraints, the Constraint Shortest Path First (CSPF) algorithm can calculate the path subject to specified constraints.

Figure 1-68 Overview of OSPF in the MPLS TE architecture



OSPF in the MPLS TE Architecture

In the MPLS TE architecture, OSPF functions as the information advertising component:

- Collects related information about TE.
- Floods TE information to devices in the same area.
- Uses the collected TE information to form the TE database (TEDB) so that CSPF can calculate routes.

OSPF does not care about information content or how MPLS uses the information.

TE-LSA

OSPF uses a new type of LSA (Type 10 opaque LSA) to collect and advertise TE information. Such LSAs contain the link status information required by TE, including the maximum link bandwidth, maximum reservable bandwidth, current reserved bandwidth, and link color. Based on the OSPF flooding mechanism, Type 10 opaque LSAs synchronize link status information among devices in an area to form a uniform TEDB for route calculation.

Interaction Between OSPF TE and CSPF

OSPF uses Type 10 LSAs to collect TE information in an area, such as the bandwidth, priority, and link metrics. After processing the collected TE information, OSPF provides it for CSPF to calculate routes.

IGP Shortcut and Forwarding Adjacency

OSPF supports IGP shortcut and forwarding adjacency. The two features allow OSPF to use a tunnel interface as an outbound interface to reach a destination.

Differences between IGP shortcut and forwarding adjacency are as follows:

- An IGP shortcut-enabled device uses a tunnel interface as an outbound interface but does not advertise the tunnel interface to neighbors. Therefore, other devices cannot use this tunnel.

- A forwarding adjacency-enabled device uses a tunnel interface as an outbound interface and advertises the tunnel interface to neighbors. Therefore, other devices can use this tunnel.
- IGP shortcut is unidirectional and needs to be configured only on the device that uses IGP shortcut.

OSPF SRLG

OSPF supports the applications of the Shared Risk Link Group (SRLG) in MPLS by obtaining information about the TE SRLG flooded among devices in an area. For details, refer to the chapter "MPLS" in this manual.

OSPF TE Tunnel Microloop Avoidance

If a network fault occurs, IGP convergence is triggered. In this case, a transient forwarding status inconsistency may occur among nodes because of their different convergence rates, which poses the risk of microloops. If the outbound interface of a route before IGP convergence is a shortcut TE tunnel interface and the OSPF TE tunnel microloop avoidance function is enabled, the outbound interface of the route remains unchanged during IGP convergence. In this case, traffic is forwarded through a hot standby TE tunnel, and the forwarding process does not depend on IGP convergence on each device, preventing microloops.

OSPF VPN

Definition

As an extension of OSPF, OSPF VPN enables Provider Edges (PEs) and Customer Edges (CEs) in VPNs to run OSPF for interworking and use OSPF to learn and advertise routes.

Purpose

As a widely used IGP, in most cases, OSPF runs in VPNs. If OSPF runs between PEs and CEs, and PEs use OSPF to advertise VPN routes to CEs, no other routing protocols need to be configured on CEs for interworking with PEs, which simplifies management and configuration of CEs.

Running OSPF Between PEs and CEs

In BGP/MPLS VPN, Multi-Protocol BGP (MP-BGP) is used to transmit routing information between PEs, whereas OSPF is used to learn and advertise routes between PEs and CEs.

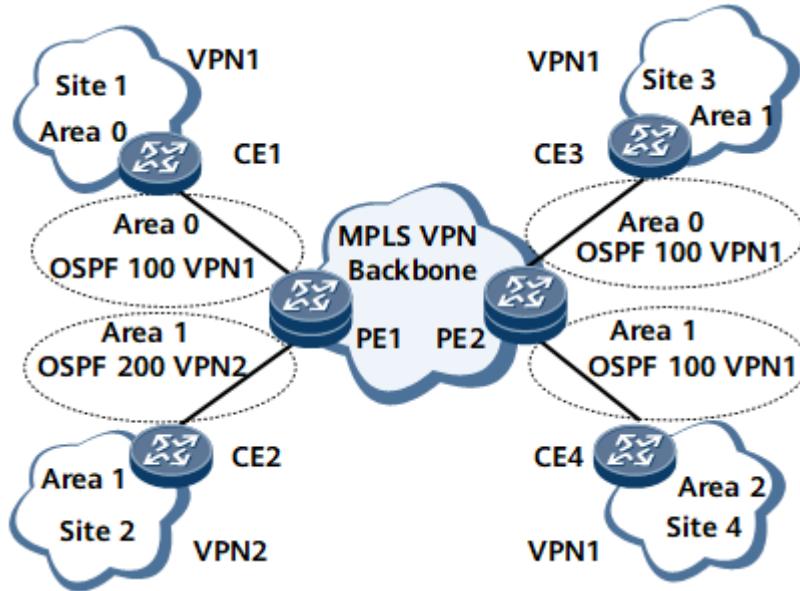
Running OSPF between PEs and CEs has the following benefits:

- OSPF is used in a site to learn routes. Running OSPF between PEs and CEs can reduce the number of the protocol types that CEs must support.
- Similarly, running OSPF both in a site and between PEs and CEs simplifies the work of network administrators and reduces the number of protocols that network administrators must be familiar with.

- When a network using OSPF but not VPN on the backbone network begins to use BGP/MPLS VPN, running OSPF between PEs and CEs facilitates the transition.

In [Figure 1-69](#), CE1, CE3, and CE4 belong to VPN 1, and the numbers following OSPF refer to the process IDs of the multiple OSPF instances running on PEs.

Figure 1-69 Networking with OSPF running between PEs and CEs



The routes that PE1 receives from CE1 are advertised to CE3 and CE4 as follows:

- PE1 imports OSPF routes of CE1 into BGP and converts them to BGP VPNv4 routes.
- PE1 uses MP-BGP to advertise the BGP VPNv4 routes to PE2.
- PE2 imports the BGP VPNv4 routes into OSPF and then advertises these routes to CE3 and CE4.

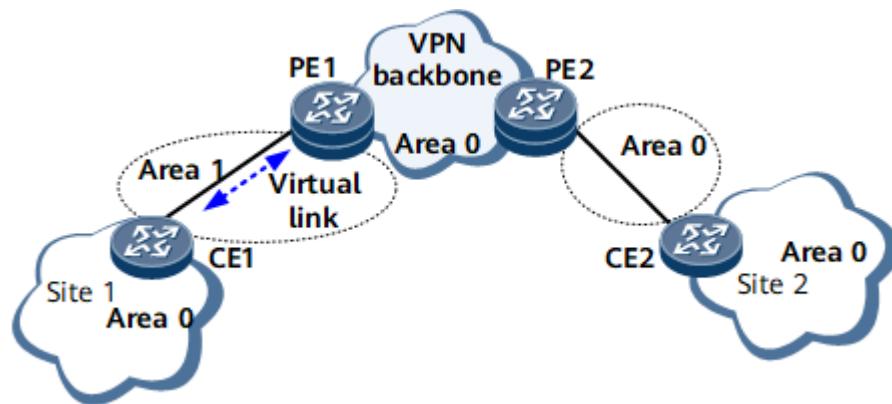
The process of advertising routes of CE4 or CE3 to CE1 is the same as the preceding process.

Configuring OSPF Areas Between PEs and CEs

OSPF areas between PEs and CEs can be non-backbone or backbone areas (Area 0). PEs can function only as ABRs.

In the extended application of OSPF VPN, the MPLS VPN backbone network serves as Area 0. OSPF requires that Area 0 be contiguous. Therefore, Area 0 of all VPN sites must be connected to the MPLS VPN backbone network. If a VPN site has OSPF Area 0, the PEs that CEs access must be connected to the backbone area of this VPN site through Area 0. If no physical link is available to directly connect PEs to the backbone area, a virtual link can be deployed between the PEs and the backbone area. [Figure 1-70](#) shows the networking for configuring OSPF areas between PEs and CEs.

Figure 1-70 Configuring OSPF areas between PEs and CEs



A non-backbone area (Area 1) is configured between PE1 and CE1, and a backbone area (Area 0) is configured in Site 1. The backbone area in Site 1 is separated from the VPN backbone area. To ensure that the backbone areas are contiguous, a virtual link is configured between PE1 and CE1.

OSPF Domain ID

If inter-area routes are advertised between local and remote OSPF areas, these areas are considered to be in the same OSPF domain.

- Domain IDs identify domains.
- Each OSPF domain has one or more domain IDs. If more than one domain ID is available, one of the domain IDs is a primary ID, and the others are secondary IDs.
- If an OSPF instance does not have a specific domain ID, its ID is considered as null.

Before advertising the remote routes sent by BGP to CEs, PEs need to determine the type of OSPF routes (Type 3, Type 5, or Type 7) to be advertised to CEs based on domain IDs.

- If local domain IDs are the same as or compatible with remote domain IDs in BGP routes, PEs advertise Type 3 routes.
- Otherwise, Type 5 or Type 7 routes are advertised.

Table 1-26 Domain ID

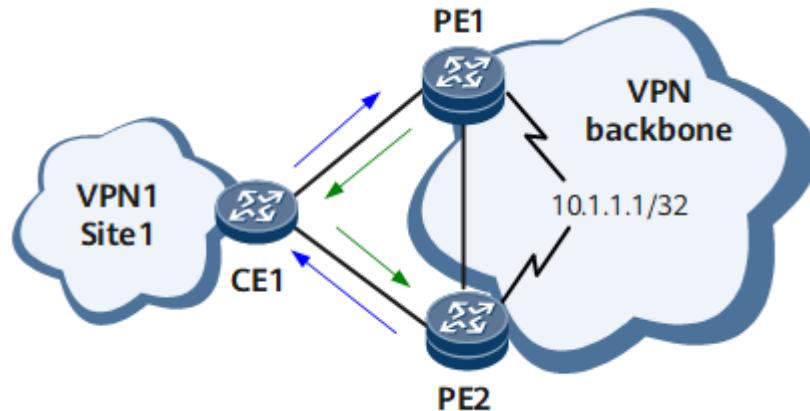
Relationship Between Local and Remote Domain IDs	Comparison Between Local and Remote Domain IDs	Type of the Generated Routes
Both the local and remote domain IDs are null.	Equal	Inter-area routes

Relationship Between Local and Remote Domain IDs	Comparison Between Local and Remote Domain IDs	Type of the Generated Routes
The remote domain ID is the same as the local primary domain ID or one of the local secondary domain IDs.	Equal	Inter-area routes
The remote domain ID is different from the local primary domain ID or any of the local secondary domain IDs.	Not equal	If the local area is a non-NSSA, external routes are generated. If the local area is an NSSA, NSSA routes are generated.

Routing Loop Prevention

Routing loops may occur between PEs and CEs when OSPF and BGP learn routes from each other.

Figure 1-71 Networking for OSPF VPN routing loops



In [Figure 1-71](#), on PE1, OSPF imports a BGP route destined for 10.1.1.1/32 and then generates and advertises a Type 5 or Type 7 LSA to CE1. Then, CE1 learns an OSPF route with 10.1.1.1/32 as the destination address and PE1 as the next hop and advertises the route to PE2. Therefore, PE2 learns an OSPF route with 10.1.1.1/32 as the destination address and CE1 as the next hop.

Similarly, CE1 also learns an OSPF route with 10.1.1.1/32 as the destination address and PE2 as the next hop. PE1 learns an OSPF route with 10.1.1.1/32 as the destination address and CE1 as the next hop.

As a result, CE1 has two equal-cost routes with PE1 and PE2 as next hops respectively, and the next hop of the routes from PE1 and PE2 to 10.1.1.1/32 is CE1, which leads to a routing loop.

In addition, the priority of an OSPF route is higher than that of a BGP route. Therefore, on PE1 and PE2, BGP routes to 10.1.1.1/32 are replaced with the OSPF route, and the OSPF route with 10.1.1.1/32 as the destination address and CE1 as the next hop is active in the routing tables of PE1 and PE2.

The BGP route is inactive, and therefore, the LSA generated when this route is imported by OSPF is deleted, which causes the OSPF route to be withdrawn. As a result, no OSPF route exists in the routing table, and the BGP route becomes active again. This cycle causes route flapping.

OSPF VPN provides a few solutions to routing loops, as described in [Table 1-27](#).

Table 1-27 Routing loop prevention measures

Feature	Definition	Function
DN-bit	It is a flag bit used by OSPF multi-instance processes to prevent routing loops.	After OSPF multi-instance is configured on the router (a PE or an MCE), the router sets the DN-bit of generated Type 3, Type 5, or Type 7 LSAs to 1 and retains the DN-bit (0) of other LSAs. When calculating routes, the OSPF multi-instance process on the router ignores the LSAs in which the DN bit is set. This prevents routing loops that occur when LSAs sent by a PE (or MCE) are sent back to the PE (or MCE) through a CE. On the network shown in Figure 1-71 , PE1 sets the DN bit in the generated Type 3, Type 5, or Type 7 LSAs to 1 and advertises these LSAs to CEs. Then, the CEs send the LSAs to PE2. Upon reception of these LSAs, PE2 checks the DN bit of the LSAs and finds that the DN bit is 1. Therefore, PE2 ignores the LSAs, which prevents routing loops.

Feature	Definition	Function
VPN Route Tag	<p>The VPN route tag is carried in Type 5 or Type 7 LSAs generated by PEs based on the received BGP VPN route.</p> <p>It is not carried in BGP extended community attributes. The VPN route tag is valid only on PEs that receive BGP routes and generate OSPF LSAs.</p>	When a PE detects that the VPN route tag in the incoming LSA is the same as the local route tag, the PE ignores the LSA, which prevents routing loops.
Default route	<p>It is a route whose destination IP address and mask are both 0.</p>	Default routes are used to forward traffic from CEs or sites where CEs reside to the VPN backbone network.

Disabling Routing Loop Prevention

NOTICE

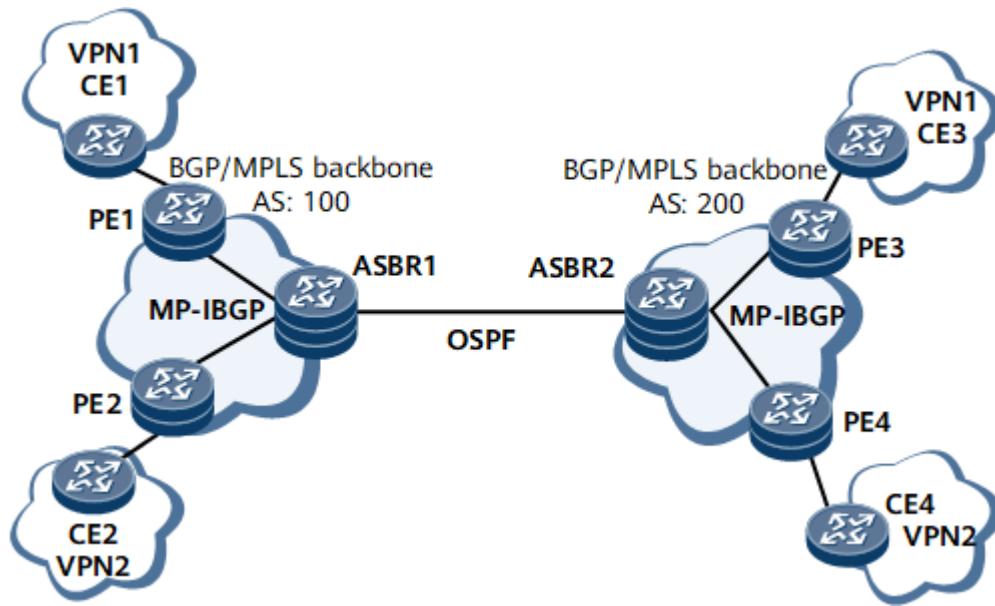
Exercise caution when disabling routing loop prevention because it may cause routing loops.

During BGP or OSPF route exchanges, routing loop prevention prevents OSPF routing loops in VPN sites.

In the inter-AS VPN Option A scenario, if OSPF runs between ASBRs to transmit VPN routes, the remote ASBR may fail to learn the OSPF routes sent by the local ASBR due to the routing loop prevention mechanism.

In [Figure 1-72](#), inter-AS VPN Option A is deployed with OSPF running between PE1 and CE1. In this example, CE1 sends VPN routes to CE3, and OSPF runs between PE1 and CE1.

Figure 1-72 Networking for inter-AS VPN Option A



1. PE1 learns routes to CE1 using the OSPF process in a VPN instance, imports these routes into MP-BGP, and sends the MP-BGP routes to ASBR1.
2. After receiving the MP-BGP routes, ASBR1 imports the routes into the OSPF process in a VPN instance and generates Type 3, Type 5, or Type 7 LSAs carrying DN bit 1.
3. ASBR2 uses OSPF to learn these LSAs and checks the DN bit of each LSA. After learning that the DN bit in each LSA is 1, ASBR2 does not add the routes carried in these LSAs to its routing table.

The routing loop prevention mechanism prevents ASBR2 from learning the OSPF routes sent from ASBR1. As a result, CE1 cannot communicate with CE3.

To address the preceding problem, use either of the following methods:

- Disable the device from setting the DN bit to 1 in the LSAs when importing BGP routes into OSPF. For example, if ASBR1 does not set the DN bit to 1 when importing MP-BGP routes into OSPF. After ASBR2 receives these routes and finds that the DN bit in the LSAs carrying these routes is 0, ASBR2 will add the routes to its routing table.
- Disable the device from checking the DN bit after receiving LSAs. For example, ASBR1 sets the DN bit to 1 in LSAs when importing MP-BGP routes into OSPF. ASBR2, however, does not check the DN bit after receiving these LSAs.

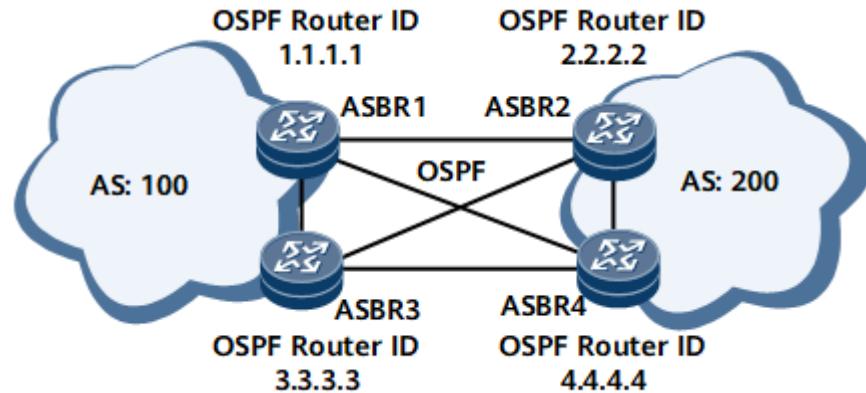
The preceding methods can be used based on specific types of LSAs. You can configure a sender to determine whether to set the DN bit to 1 or configure a receiver to determine whether to check the DN bit in the Type 3 LSAs based on the router ID of the device that generates the Type 3 LSAs.

In the inter-AS VPN Option A scenario shown in [Figure 1-73](#), the four ASBRs are fully meshed and run OSPF. ASBR2 may receive the Type 3, Type 5, or Type 7 LSAs generated on ASBR4. If ASBR2 is not configured to check the DN bit in the LSAs,

ASBR2 will accept the Type 3 LSAs, which may cause routing loops, as described in [Routing Loop Prevention](#). ASBR2 will deny the Type 5 or Type 7 LSAs, because the VPN route tags carried in the LSAs are the same as the default VPN route tag of the OSPF process on ASBR2.

To address the routing loop problem caused by Type 3 LSAs, ASBR2 can be disabled from checking the DN bit in the Type 3 LSAs generated by devices with router ID 1.1.1.1 and router ID 3.3.3.3. After the configuration is complete, if ASBR2 receives Type 3 LSAs sent by ASBR4 with router ID 4.4.4.4, ASBR2 checks the DN bit and denies these Type 3 LSAs because the DN bit is set to 1.

Figure 1-73 Networking for fully meshed ASBRs in the inter-AS VPN Option A scenario

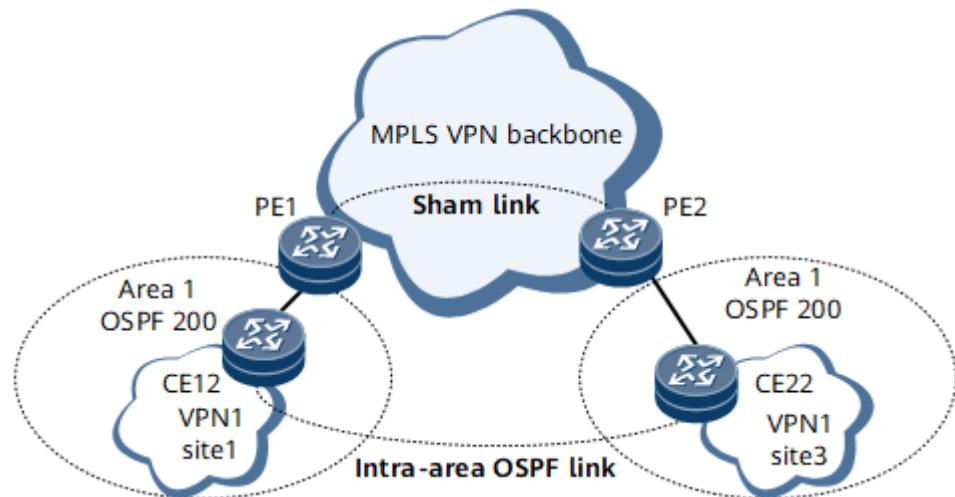


Sham Link

OSPF sham links are unnumbered P2P links between two PEs over an MPLS VPN backbone network.

Generally, BGP extended community attributes carry routing information over the MPLS VPN backbone between BGP peers. OSPF running on the other PE can use the routing information to generate inter-area routes from PEs to CEs.

Figure 1-74 OSPF Sham link



In [Figure 1-74](#), if an intra-area OSPF link exists between the network segments of local and remote CEs. Routes that pass through the intra-area route link and have higher priorities than inter-area routes that pass through the MPLS VPN backbone network. As a result, VPN traffic is always forwarded through the intra-area route instead of the backbone network. To prevent such a problem, an OSPF sham link can be established between PEs so that the routes that pass through the MPLS VPN backbone network also become OSPF intra-area routes and take precedence.

- A sham link is a link between two VPN instances. Each VPN instance contains the address of an end-point of a sham link. The address is a loopback address with the 32-bit mask in the VPN address space on the PE.
- After a sham link is established between two PEs, the PEs become neighbors on the sham link and exchange routing information.
- A sham link functions as a P2P link within an area. Users can select a route from the sham link and intra-area route link by adjusting the metric.

Multi-VPN-Instance CE

OSPF multi-instance generally runs on PEs. Devices that run OSPF multi-instance within user LANs are called Multi-VPN-Instance CEs (MCEs).

Compared with OSPF multi-instance running on PEs, MCEs have the following characteristics:

- MCEs do not need to support OSPF-BGP association.
- MCEs establish one OSPF instance for each service. Different virtual CEs transmit different services, which ensures LAN security at a low cost.
- MCEs implement different OSPF instances on a CE. The key to implementing MCEs is to disable loop detection and calculate routes directly. MCEs also use the received LSAs with the DN-bit 1 for route calculation.

OSPF NSSA

Background

As defined in OSPF, stub areas cannot import external routes. This prevents a large number of external routes from consuming router bandwidth and storage resources in the stub areas. If external routes need to be imported and resource consumption caused by external routes needs to be avoided, stub areas cannot meet the requirements. In this case, not-so-stubby areas (NSSAs) can be configured.

OSPF NSSA is a special type of area. There are many similarities between NSSAs and stub areas. However, different from stub areas, NSSAs can import AS external routes into the OSPF AS and advertise the imported routes in the OSPF AS without learning external routes from other areas on the OSPF network.

Related Concepts

- **N-bit**

A router uses the N-bit carried in a Hello packet to identify the area type that it supports. The same area type must be configured for all routers in an area. If routers have different area types, they cannot establish OSPF neighbor

relationships. Some non-Huawei devices set the N-bit in OSPF DD packets. To interwork with these non-Huawei devices, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X can be configured to be compatible with these non-Huawei devices.

- **Type7 LSA**

Type 7 LSAs, which describe imported external routes, are introduced to support NSSAs. Type 7 LSAs are generated by an ASBR in an NSSA and advertised only within the NSSA. After an ABR in an NSSA receives Type 7 LSAs, it selectively translates Type 7 LSAs into Type 5 LSAs to advertise external routes to other areas on an OSPF network.

Principles

To advertise external routes imported by an NSSA to other areas, a translator must translate Type 7 LSAs into Type 5 LSAs. [Figure 1-75](#) shows the translation process.

- The propagate bit (P-bit) is used to notify a translator whether Type 7 LSAs need to be translated.
- By default, the translator is the ABR with the largest router ID in the NSSA.
- Only Type 7 LSAs with the P-bit set and a non-zero forwarding address (FA) can be translated into Type 5 LSAs. An FA indicates the address to which packets to a destination address will be forwarded.

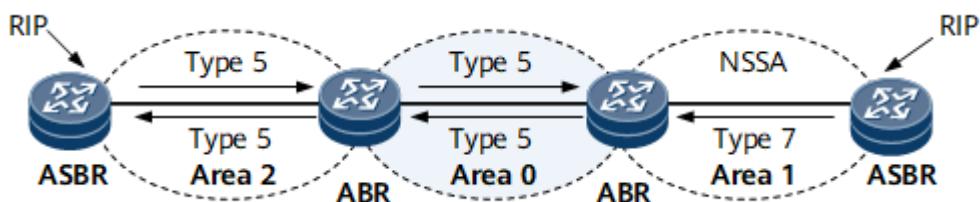
NOTE

An FA indicates the address to which packets to a destination address will be forwarded.

The loopback interface address in an area is preferentially selected as the FA. If no loopback interface exists, the address of the interface that is Up and has the smallest logical index in the area is selected as the FA.

- The P-bit is not set for default routes in Type 7 LSAs generated by an ABR.

Figure 1-75 NSSA



Advantages

Multiple ABRs may be deployed in an NSSA. To prevent routing loops caused by default routes, ABRs do not calculate the default routes advertised by each other.

OSPF Local MT

Background

When multicast and an MPLS TE tunnel are configured on a network and the TE tunnel is configured with IGP Shortcut, the outbound interface of the route

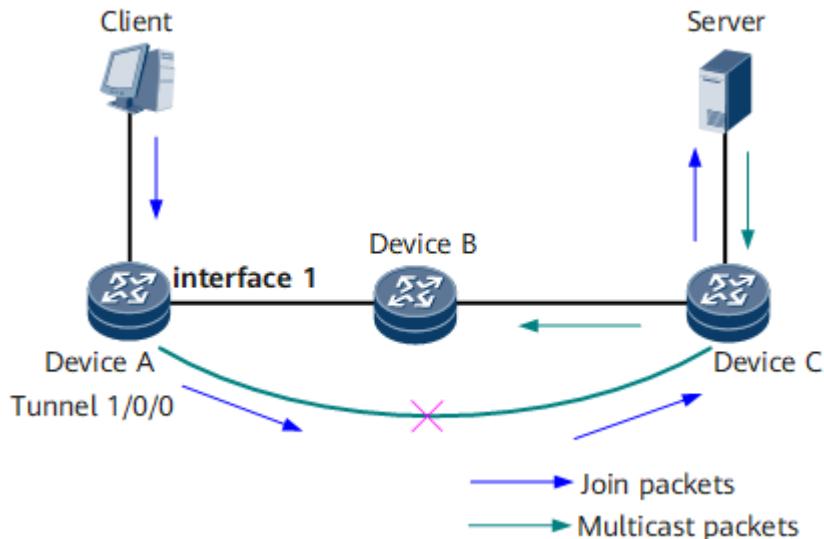
calculated by an IGP may be not an actual physical interface but a TE tunnel interface. The TE tunnel interface on the Device sends multicast Join messages over a unicast route to the multicast source address. The multicast Join messages are transparent to the Device spanned by the TE tunnel. As a result, the Device spanned by the TE tunnel cannot generate multicast forwarding entries.

To resolve the problem, configure OSPF local multicast topology (MT) to create a multicast routing table for multicast packet forwarding.

Implementation

Multicast and an MPLS TE tunnel are deployed on the network, and the TE tunnel is enabled with IGP Shortcut. As shown in [Figure 1-76](#), DeviceB is spanned by the TE tunnel and therefore does not create any multicast forwarding entry.

Figure 1-76 OSPF Local MT



Because the TE tunnel is unidirectional, multicast data packets sent from the multicast source are directly sent to the routers spanned by the tunnel through physical interfaces. These routers, however, do not have multicast forwarding entries. As a result, the multicast data packets are discarded, and services are unavailable.

After local MT is enabled, if the outbound interface of the calculated route is an IGP Shortcut TE tunnel interface, the route management (RM) module creates a separate Multicast IGP (MIGP) routing table for the multicast protocol, calculates the actual physical outbound interface for the route, and then adds the route to the MIGP routing table. Multicast then uses routes in the MIGP routing table to forward packets.

In [Figure 1-76](#), after the messages requesting to join a multicast group reach DeviceA, they are forwarded to DeviceB through interface 1. In this manner, DeviceB can correctly create the multicast forwarding table.

BFD for OSPF

Definition

Bidirectional Forwarding Detection (BFD) is a mechanism to detect communication faults between forwarding engines.

To be specific, BFD detects the connectivity of a data protocol along a path between two systems. The path can be a physical link, a logical link, or a tunnel.

In BFD for OSPF, a BFD session is associated with OSPF. The BFD session quickly detects a link fault and then notifies OSPF of the fault, which speeds up OSPF's response to network topology changes.

Purpose

A link fault or a topology change causes routers to recalculate routes. Routing protocol convergence must be as quick as possible to improve network availability. Link faults are inevitable, and therefore a solution must be provided to quickly detect faults and notify routing protocols.

BFD for Open Shortest Path First (OSPF) associates BFD sessions with OSPF. After BFD for OSPF is configured, BFD quickly detects link faults and notifies OSPF of the faults. BFD for OSPF accelerates OSPF response to network topology changes.

Table 1-28 describes OSPF convergence speeds before and after BFD for OSPF is configured.

Table 1-28 OSPF convergence speeds before and after BFD for OSPF is configured

Item	Link Fault Detection Mechanism	Convergence Speed
BFD for OSPF is not configured.	An OSPF Dead timer expires.	Second-level
BFD for OSPF is configured.	A BFD session goes Down.	Millisecond-level

Principles

Figure 1-77 BFD for OSPF

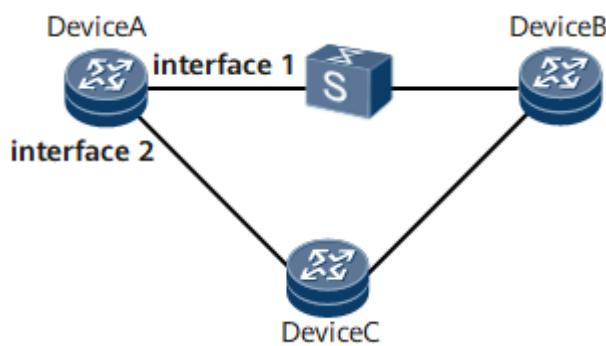


Figure 1-77 shows a typical network topology with BFD for OSPF configured. The principles of BFD for OSPF are described as follows:

1. OSPF neighbor relationships are established between these three routers.
2. After a neighbor relationship becomes Full, a BFD session is established.
3. The outbound interface on Device A connected to Device B is interface 1. If the link between Device A and Device B fails, BFD detects the fault and then notifies Device A of the fault.
4. Device A processes the event that a neighbor relationship goes Down and recalculates routes. The new route passes through Device C and reaches Device A, with interface 2 as the outbound interface.

OSPF GTSM

Definition

Generalized TTL security mechanism (GTSM) is a mechanism that protects services over the IP layer by checking whether the TTL value in an IP packet header is within a pre-defined range.

Purpose

On networks, attackers may simulate OSPF packets and keep sending them to a device. After receiving these packets, the device directly sends them to the control plane for processing without checking their validity if the packets are destined for the device. As a result, the control plane is busy processing these packets, resulting in high CPU usage.

GTSM is used to protect the TCP/IP-based control plane against CPU-utilization attacks, such as CPU-overload attacks.

Principles

GTSM-enabled devices check the TTL value in each received packet based on a configured policy. The packets that fail to pass the policy are discarded or sent to the control plane, which prevents the devices from possible CPU-utilization attacks. A GTSM policy involves the following items:

- Source address of the IP packet sent to the device
- VPN instance to which the packet belongs
- Protocol number of the IP packet (89 for OSPF)
- Source port number and destination port number of protocols above TCP/UDP
- Valid TTL range

GTSM is implemented as follows:

- For directly connected OSPF neighbors, the TTL value of the unicast protocol packets to be sent is set to 255.
- For multi-hop neighbors, a reasonable TTL range is defined.

The applicability of GTSM is as follows:

- GTSM takes effect on unicast packets rather than multicast packets. This is because the TTL value of multicast packets can only be 255, and therefore GTSM is not needed to protect against multicast packets.
- GTSM does not support tunnel-based neighbors.

OSPF Smart-discover

Definition

Hello packets are periodically sent on OSPF interfaces of routers. By exchanging Hello packets, the routers establish and maintain the neighbor relationship, and elect the DR and the Backup Designated Router (BDR) on the multiple-access network (broadcast or NBMA network). OSPF uses a Hello timer to control the interval at which Hello packets are sent. A router can send Hello packets again only after the Hello timer expires. Neighbors keep waiting to receive Hello packets until the Hello timer expires. This process delays the establishment of OSPF neighbor relationships or election of the DR and the BDR.

Enabling Smart-discover can solve the preceding problem.

Table 1-29 Processing differences with and without Smart-discover

With or Without Smart-discover	Processing
Without Smart-discover	<ul style="list-style-type: none">• Hello packets are sent only when the Hello timer expires.• Hello packets are sent at the Hello interval.• Neighbors keep waiting to receive Hello packets within the Dead interval.
With Smart-discover	<ul style="list-style-type: none">• Hello packets are sent directly regardless of whether the Hello timer expires.• Neighbors can receive packets and change the state immediately.

Principles

In the following situations, Smart-discover-enabled interfaces can send Hello packets to neighbors regardless of whether the Hello timer expires:

- On broadcast or NBMA networks, neighbor relationships can be established and a DR and a BDR can be elected rapidly.
 - The neighbor status becomes 2-way for the first time or returns to Init from 2-way or a higher state.
 - The interface status of the DR or BDR on a multiple-access network changes.
- On P2P or P2MP networks, neighbor relationships can be established rapidly. The establishment of neighbor relationships on a P2P or P2MP network is the same as that on a broadcast or NBMA network.

OSPF-BGP Synchronization

Background

When a new device is deployed on a network or a device is restarted, network traffic may be lost during BGP route convergence because IGP routes converge more quickly than BGP routes.

OSPF-BGP synchronization can address this problem.

Purpose

If a backup link exists, BGP traffic may be lost during traffic switchback because BGP routes converge more slowly than OSPF routes do.

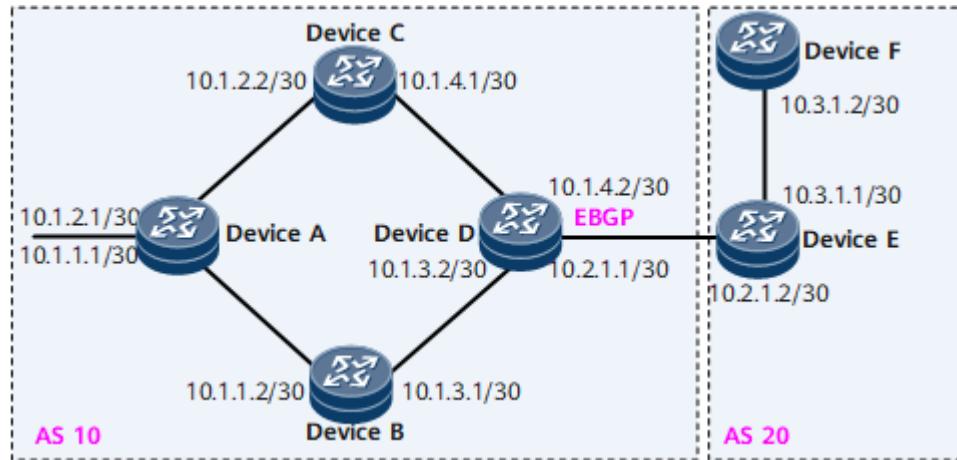
In **Figure 1-78**, Device A, Device B, Device C, and Device D run OSPF and establish IBGP connections. Device C functions as the backup of Device B. When the network is stable, BGP and OSPF routes converge completely on the router.

In most cases, traffic from Device A to 10.3.1.0/30 passes through Device B. If Device B fails, traffic is switched to Device C. After Device B recovers, traffic is switched back to Device B. During this process, packet loss occurs.

Consequently, convergence of OSPF routes is complete whereas BGP route convergence is still going on. As a result, Device B does not have the route to 10.3.1.0/30.

When packets from Device A to 10.3.1.0/30 reach Device B, Device B discards them because Device B does not have the route to 10.3.1.0/30.

Figure 1-78 Networking for OSPF-BGP synchronization



Principles

If OSPF-BGP synchronization is configured on a device, the device remains as a stub router during the set synchronization period. During this period, the link metric in the LSA advertised by the device is set to the maximum value (65535), instructing other OSPF routers not to use it as a transit router for data forwarding.

In [Figure 1-78](#), OSPF-BGP synchronization is enabled on Router B. In this situation, before BGP route convergence is complete, Device A keeps forwarding data through Device C rather than Device B until BGP route convergence on Device B is complete.

LDP-IGP Synchronization

Background

LDP-IGP synchronization is used to synchronize the status between LDP and an IGP to minimize the traffic loss time if a network fault triggers the LDP and IGP switching.

On a network with active and standby links, if the active link fails, IGP routes and an LSP are switched to the standby link. After the active link recovers, IGP routes are switched back to the active link before LDP convergence is complete. In this case, the LSP along the active link takes time to make preparations, such as adjacency restoration, before being established. As a result, LSP traffic is discarded. If an LDP session or adjacency between nodes fails on the active link, the LSP along the active link is deleted. However, the IGP still uses the active link, and as a result, LSP traffic cannot be switched to the standby link, and is continuously discarded.

NOTE

LDP-IGP synchronization supports OSPFv2 and IS-IS (IPv4).

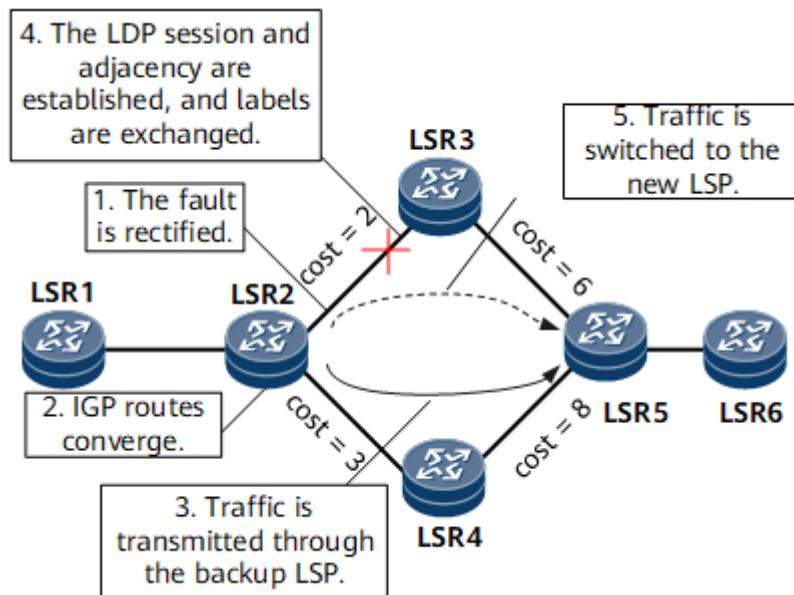
On a network enabled with LDP-IGP synchronization, an IGP keeps advertising the maximum cost of an IGP route over the new active link to delay IGP route convergence until LDP converges. That is, before the LSP of the active link is established, the LSP of the standby link is retained so that the traffic continues to be forwarded through the standby link. The standby LSP is torn down only after the active LSP is established successfully.

LDP-IGP synchronization involves the following timers:

- Hold-max-cost timer
- Delay timer

Implementation

Figure 1-79 Switchback in LDP-IGP synchronization



- The network shown in **Figure 1-79** has active and standby links. When the active link recovers from a fault, traffic is switched from the standby link back to the active link. During the traffic switchback, the standby LSP cannot be used, and a new LSP cannot be set up over the active link once IGP route convergence is complete. This causes a traffic interruption for a short period of time. To prevent this problem, LDP-IGP synchronization can be configured to delay IGP route switchback until LDP convergence is complete. Before convergence of the active LSP completes, the standby LSP is retained, so that the traffic continues to be forwarded through the standby LSP until the active LSP is successfully established. Then the standby LSP is torn down. The detailed process is as follows:
 - a. The link fault is rectified.
 - b. An LDP session is set up between LSR2 and LSR3. The IGP advertises the maximum cost of the active link to delay the IGP route switchback.
 - c. Traffic is still forwarded along the standby LSP.
 - d. The LDP session is set up. Label messages are exchanged to notify the IGP to start synchronization.
 - e. The IGP advertises the normal cost of the active link, and its routes converge to the original forwarding path. The LSP is reestablished and entries are delivered to the forwarding table (within milliseconds).
- If an LDP session between nodes fails on the active link, the LSP along the active link is deleted. However, the IGP still uses the active link, and as a result, LSP traffic cannot be switched to the standby link, and is continuously discarded. To prevent this problem, you can configure LDP-IGP synchronization. If an LDP session fails, LDP notifies an IGP of the failure. The IGP advertises the maximum cost of the failed link, which enables the route to switch from the active link to the standby link. In addition to the LSP

switchover from the primary LSP to the backup LSP, LDP-IGP synchronization is implemented. The process of LDP-IGP synchronization is as follows:

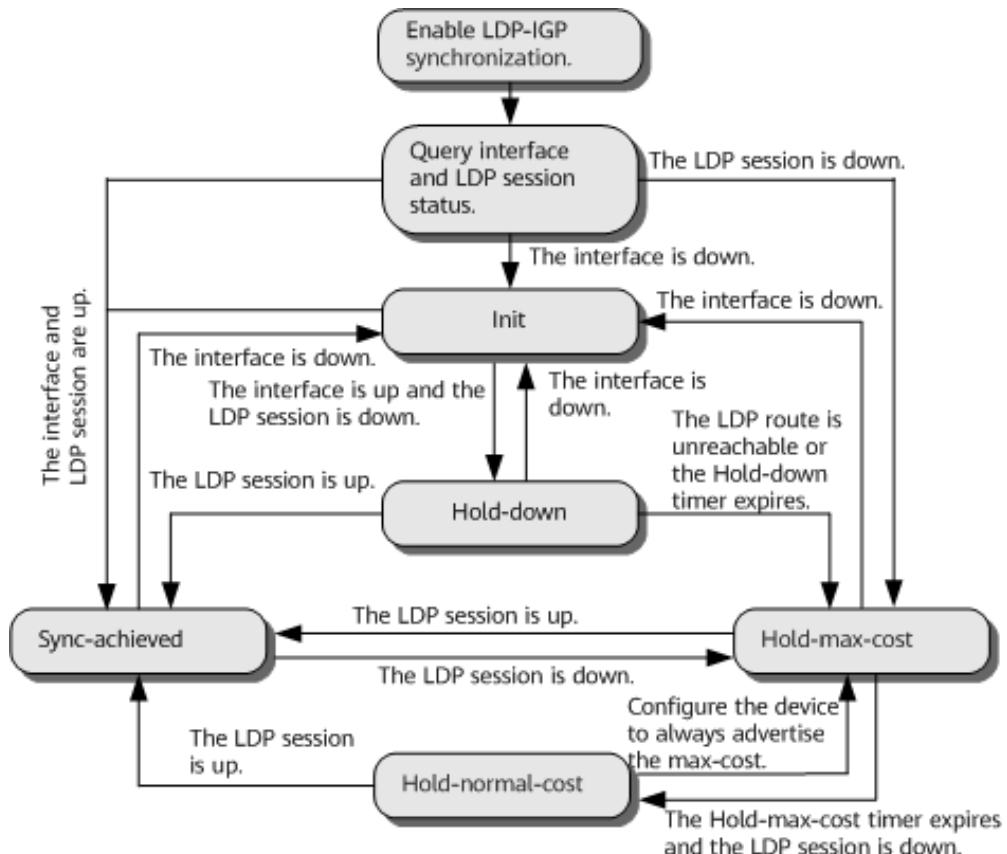
- a. An LDP session between two nodes on the active link fails.
- b. LDP notifies the IGP of the failure in the session over the active link. The IGP then advertises the maximum cost along the active link.
- c. The IGP route switches to the standby link.
- d. An LSP is set up over the standby link, and then forwarding entries are delivered.

To prevent repeated failures in LDP session reestablishment, you can use the Hold-max-cost timer to configure the device to always advertise the maximum cost, so that traffic is transmitted along the standby link before the LDP session is reestablished on the active link.

- LDP-IGP synchronization state transition mechanism

After LDP-IGP synchronization is enabled on an interface, the IGP queries the status of the interface and LDP session according to the process shown in **Figure 1-80**, enters the corresponding state according to the query result, and then transits the state according to **Figure 1-80**.

Figure 1-80 LDP-IGP synchronization state transition



NOTE

The preceding states may slightly vary between different IGPs.

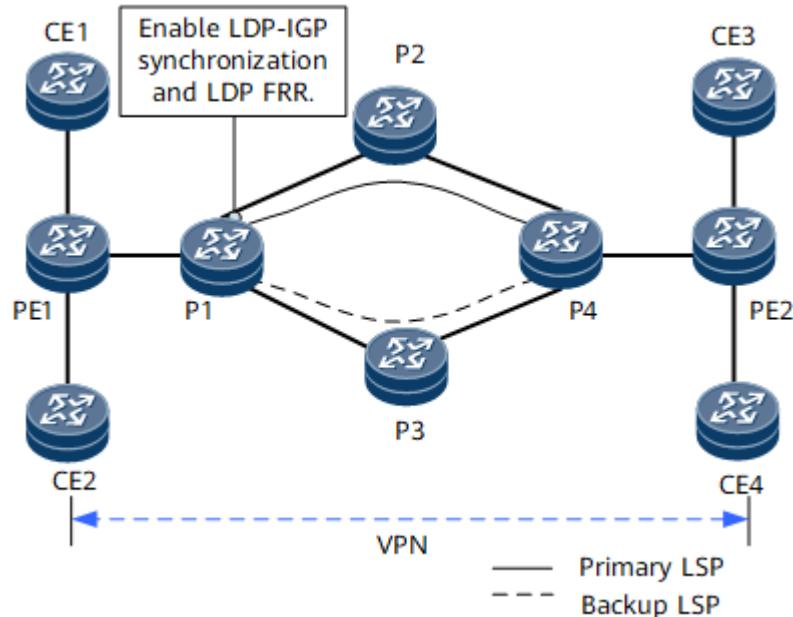
- When OSPF is used as the IGP, the state transition is the same as that in [Figure 1-80](#).
- When IS-IS is used as the IGP, the Hold-normal-cost state does not exist. After the Hold-max-cost timer expires, IS-IS advertises the normal link cost, but the Hold-max-cost state is displayed even though this state is nonexistent.

Usage Scenario

LDP-IGP synchronization applies to the following scenario:

On the network shown in [Figure 1-81](#), an active link and a standby link are established. LDP-IGP synchronization and LDP FRR are deployed.

Figure 1-81 LDP-IGP synchronization scenario



Benefits

Packet loss is reduced during an active/standby link switchover, improving network reliability.

OSPF Fast Convergence

OSPF fast convergence is an extended feature of OSPF to speed up the convergence of routes. It includes the following components:

- Partial Route Calculation (PRC): calculates only those routes which have changed when the network topology changes.
- An OSPF intelligent timer: can dynamically adjust its value based on the user's configuration and the interval at which an event is triggered, such as the route calculation interval, which ensures rapid and stable network operation.

OSPF intelligent timer uses the exponential backoff technology so that the value of the timer can reach the millisecond level.

PRC

When a node in a network topology changes, the Dijkstra algorithm needs to recalculate all routes on the network. This calculation takes a long time and consumes a large number of CPU resources, which affects the convergence speed on the entire network. However, PRC uses only the nodes that have changed to recalculate routes, thereby decreasing CPU usage.

In route calculation, a leaf represents a route, and a node represents a device. Either an SPT change or a leaf change causes a routing information change. The SPT change is irrelevant to the leaf change. PRC processes routing information as follows:

- If the SPT changes, PRC calculates all the leaves only on the changed node.
- If the SPT remains unchanged, PRC calculates only the changed leaves.

For example, if a new route is imported, the SPT of the entire network remains unchanged. In this case, PRC updates only the interface route for this node, thereby reducing the CPU usage.

OSPF Intelligent Timer

On an unstable network, routes are calculated frequently, which consumes a great number of CPU resources. In addition, link-state advertisements (LSAs) that describe the unstable topology are generated and transmitted on the unstable network. Frequently processing such LSAs affects the rapid and stable operation of the entire network.

To speed up route convergence on the entire network, the OSPF intelligent timer controls route calculation, LSA generation, and LSA receiving.

The OSPF intelligent timer works as follows:

- On a network where routes are calculated repeatedly, the OSPF intelligent timer dynamically adjusts the route calculation based on user's configuration and the exponential backoff technology. The number of route calculation times and the CPU resource consumption are decreased. Routes are calculated after the network topology stabilizes.
- On an unstable network, if a router generates or receives LSAs due to frequent topology changes, the OSPF intelligent timer can dynamically adjust the interval. No LSAs are generated or processed within an interval, which prevents invalid LSAs from being generated and advertised on the entire network.

OSPF Neighbor Relationship Flapping Suppression

OSPF neighbor relationship flapping suppression works by delaying OSPF neighbor relationship reestablishment or setting the link cost to the maximum value.

Background

If an interface carrying OSPF services frequently alternates between up and down, frequent neighbor relationship flapping will occur. During the flapping, OSPF

frequently sends Hello packets to reestablish the neighbor relationship, synchronizes LSDBs, and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, OSPF services, and other OSPF-dependent services, such as LDP and BGP. OSPF neighbor relationship flapping suppression can address this problem by delaying OSPF neighbor relationship reestablishment or preventing service traffic from passing through flapping links.

Related Concepts

Flapping-event: reported when the status of a neighbor relationship on an interface last changes from Full to a non-Full state. The flapping-event triggers flapping detection.

Flapping-count: number of times flapping has occurred.

Detecting-interval: detection interval. The interval is used to determine whether to trigger a valid flapping_event.

Threshold: flapping suppression threshold. When the flapping_count reaches or exceeds **threshold**, flapping suppression takes effect.

Resume-interval: interval for exiting neighbor relationship flapping suppression. If the interval between two successive valid flapping_events is longer than **resume-interval**, the flapping_count is reset.

Implementation

Flapping detection

Each OSPF interface on which OSPF neighbor relationship flapping suppression is enabled starts a flapping counter. If the interval between two successive neighbor status changes from Full to a non-Full state is shorter than **detecting-interval**, a valid flapping_event is recorded, and the flapping_count increases by 1. When the flapping_count reaches or exceeds **threshold**, flapping suppression takes effect. If the interval between two successive neighbor status changes from Full to a non-Full state is longer than **resume-interval**, the flapping_count is reset.

The **detecting-interval**, **threshold**, and **resume-interval** are configurable.



The value of **resume-interval** must be greater than that of **detecting-interval**.

Flapping suppression

Flapping suppression works in either Hold-down or Hold-max-cost mode.

- Hold-down mode: In the case of frequent flooding and topology changes during neighbor relationship establishment, interfaces prevent neighbor relationship reestablishment during Hold-down suppression, which minimizes LSDB synchronization attempts and packet exchanges.
- Hold-max-cost mode: In this mode, the link cost is set to the maximum value (65535) within a period of time to prevent service traffic from passing through flapping links.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

By default, the Hold-max-cost mode takes effect. The mode and suppression duration can be changed manually.

If an attack causes frequent neighbor relationship flapping, Hold-down mode can minimize the impact of the attack.

NOTE

When an interface enters the flapping suppression state, all neighbor relationships on the interface enter the state accordingly.

Exiting from flapping suppression

Interfaces exit from flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding OSPF process is reset.
- An OSPF neighbor is reset.
- A command is run to exit from flapping suppression.

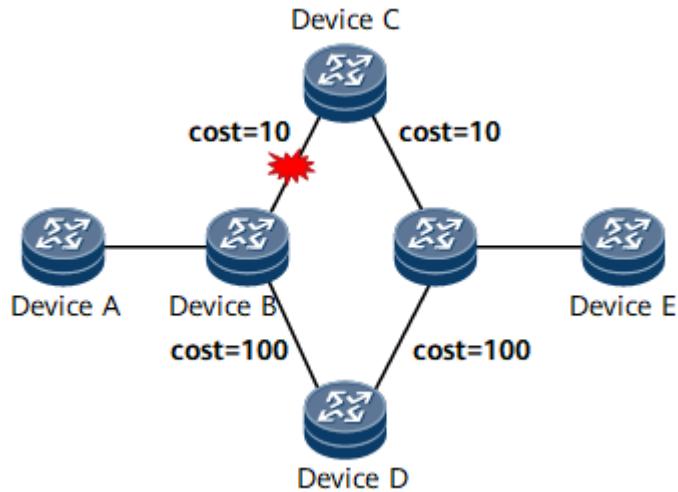
Typical Scenarios

Basic scenario

In [Figure 1-82](#), the traffic forwarding path is Device A -> Device B -> Device C -> Device E before a link failure occurs. After the link between Device B and Device C fails, the forwarding path switches to Device A -> Device B -> Device D -> Device E. If the neighbor relationship between Device B and Device C frequently flaps at the early stage of the path switchover, the forwarding path will be switched frequently, causing traffic loss and affecting network stability. If the neighbor relationship flapping meets suppression conditions, flapping suppression takes effect.

- If flapping suppression works in Hold-down mode, the neighbor relationship between Device B and Device C is prevented from being reestablished during the suppression period, in which traffic is forwarded along the path Device A -> Device B -> Device D -> Device E.
- If flapping suppression works in Hold-max-cost mode, 65535 is used as the cost of the link between Device B and Device C during the suppression period, and traffic is forwarded along the path Device A -> Device B -> Device D -> Device E.

Figure 1-82 Flapping suppression in a basic scenario



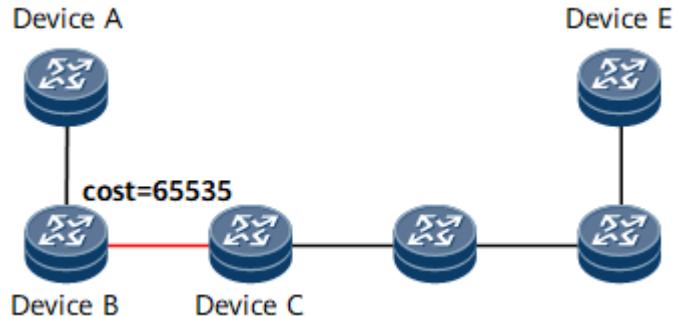
Single-forwarding path scenario

When only one forwarding path exists on the network, the flapping of the neighbor relationship between any two devices on the path will interrupt traffic forwarding. In [Figure 1-83](#), the traffic forwarding path is Device A -> Device B -> Device C -> Device E. If the neighbor relationship between Device B and Device C flaps, and the flapping meets suppression conditions, flapping suppression takes effect. However, if the neighbor relationship between Device B and Device C is prevented from being reestablished, the whole network will be divided. Therefore, Hold-max-cost mode (rather than Hold-down mode) is recommended. If flapping suppression works in Hold-max-cost mode, 65535 is used as the cost of the link between Device B and Device C during the suppression period. After the network stabilizes and the suppression timer expires, the link is restored.

NOTE

By default, the Hold-max-cost mode takes effect.

Figure 1-83 Flapping suppression in a single-forwarding path scenario

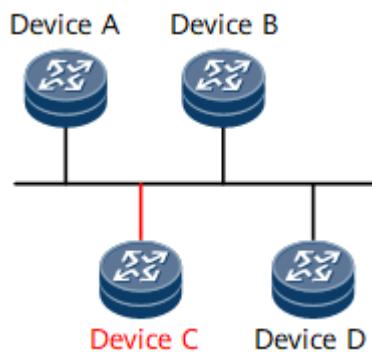


Broadcast scenario

In [Figure 1-84](#), four devices are deployed on the same broadcast network using switches, and the devices are broadcast network neighbors. If Device C flaps due to a link failure, and Device A and Device B were deployed at different time

(Device A was deployed earlier for example) or the flapping suppression parameters on Device A and Device B are different, Device A first detects the flapping and suppresses Device C. Consequently, the Hello packets sent by Device A do not carry Device C's router ID. However, Device B has not detected the flapping yet and still considers Device C a valid node. As a result, the DR candidates identified by Device A are Device B and Device D, whereas the DR candidates identified by Device B are Device A, Device C, and Device D. Different DR candidates result in a different DR election result, which may lead to route calculation errors. In scenarios where an interface has multiple neighbors, such as on a broadcast, P2MP, or NBMA network, all neighbors on the interface are suppressed if the last flapping event that the neighbor status on the interface becomes ExStart or Down is detected (flapping detection cannot be performed by neighbor). Specifically, if Device C flaps, Device A, Device B, and Device D on the broadcast network are all suppressed. After the network stabilizes and the suppression timer expires, Device A, Device B, and Device D are restored to normal status.

Figure 1-84 Flapping suppression on a broadcast network



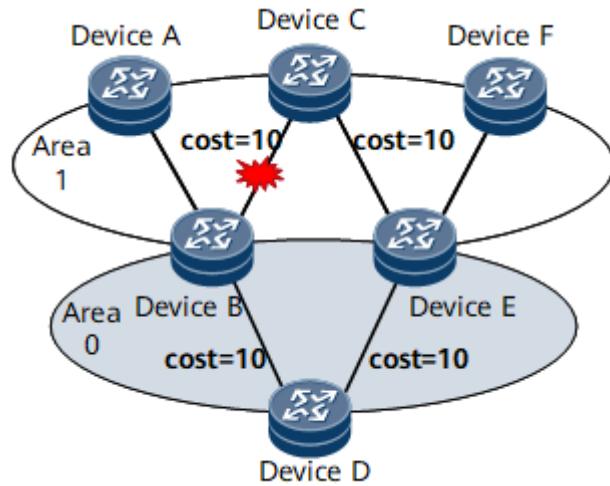
Multi-area scenario

In **Figure 1-85**, Device A, Device B, Device C, Device E, and Device F are connected in area 1, and Device B, Device D, and Device E are connected in backbone area 0. Traffic from Device A to Device F is preferentially forwarded along an intra-area route, and the forwarding path is Device A -> Device B -> Device C -> Device E -> Device F. If the neighbor relationship between Device B and Device C flaps and the flapping meets suppression conditions, flapping suppression takes effect in the default mode (Hold-max-cost). However, the forwarding path remains unchanged (Device A -> Device B -> Device C -> Device E -> Device F) after the neighbor flapping occurs because intra-area routes take precedence over inter-area routes during route selection regardless of costs according to OSPF route selection rules. The Hold-max-cost mode cannot suppress traffic path switching in this case. To prevent traffic loss in multi-area scenarios, configure Hold-down mode to prevent the neighbor relationship between Device B and Device C from being reestablished during the suppression period. During this period, traffic is forwarded along the path Device A -> Device B -> Device D -> Device E -> Device F.

NOTE

By default, the Hold-max-cost mode takes effect. The mode can be changed to Hold-down manually.

Figure 1-85 Flapping suppression in a multi-area scenario



Scenario with both LDP-IGP synchronization and OSPF neighbor relationship flapping suppression configured

In [Figure 1-86](#), if the link between PE1 and P1 fails, an LDP LSP switchover is implemented immediately, causing the original LDP LSP to be deleted before a new LDP LSP is established. To prevent traffic loss, LDP-IGP synchronization needs to be configured. With LDP-IGP synchronization, 65535 is used as the cost of the new LSP to be established. After the new LSP is established, the original cost takes effect. Consequently, the original LSP is deleted, and LDP traffic is forwarded along the new LSP.

LDP-IGP synchronization and OSPF neighbor relationship flapping suppression work in either Hold-down or Hold-max-cost mode. If both functions are configured, Hold-down mode takes precedence over Hold-max-cost mode, followed by the configured link cost. [Table 1-30](#) lists the suppression modes that take effect in different situations.

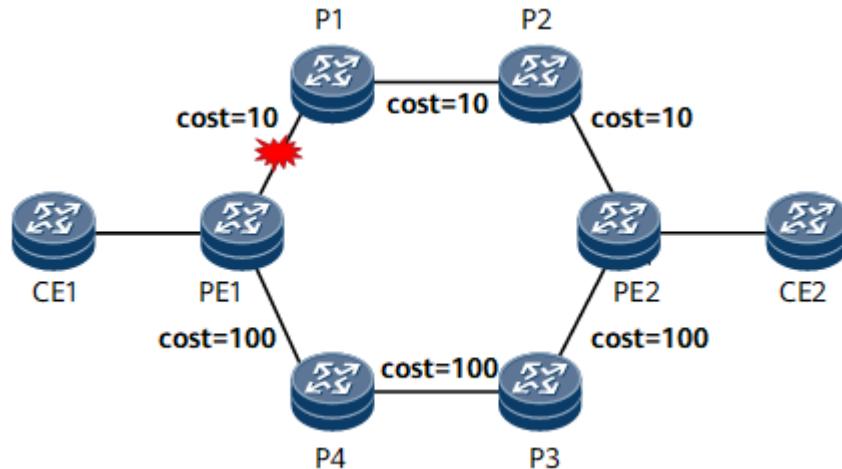
Table 1-30 Principles for selecting the suppression modes that take effect in different situations

LDP-IGP Synchronization/OSPF Neighbor Relationship Flapping Suppression Mode	LDP-IGP Synchronization Hold-down Mode	LDP-IGP Synchronization Hold-max-cost Mode	Exited from LDP-IGP Synchronization Suppression
OSPF Neighbor Relationship Flapping Suppression Hold-down Mode	Hold-down	Hold-down	Hold-down

LDP-IGP Synchronization/OSPF Neighbor Relationship Flapping Suppression Mode	LDP-IGP Synchronization Hold-down Mode	LDP-IGP Synchronization Hold-max-cost Mode	Exited from LDP-IGP Synchronization Suppression
OSPF Neighbor Relationship Flapping Suppression Hold-max-cost Mode	Hold-down	Hold-max-cost	Hold-max-cost
Exited from OSPF Neighbor Relationship Flapping Suppression	Hold-down	Hold-max-cost	Exited from LDP-IGP synchronization and OSPF neighbor relationship flapping suppression

For example, the link between PE1 and P1 frequently flaps in [Figure 1-86](#), and both LDP-IGP synchronization and OSPF neighbor relationship flapping suppression are configured. In this case, the suppression mode is selected based on the preceding principles. No matter which mode (Hold-down or Hold-max-cost) is selected, the forwarding path is PE1 -> P4 -> P3 -> PE2.

Figure 1-86 Scenario with both LDP-IGP synchronization and OSPF neighbor relationship flapping suppression configured



Scenario with both bit-error-triggered protection switching and OSPF neighbor relationship flapping suppression configured

Bit-error-triggered protection switching is used to protect link quality. If link quality detection detects that the link quality is poor and the bit error rate (BER) is high, a bit error event is reported. User services carried on the link with a high BER are adversely affected. Therefore, user traffic needs to be switched to other links. After receiving a bit error event, OSPF adjusts the interface cost to the maximum value (65535). Then, OSPF recalculates routes and switch service traffic to the backup link. If both bit-error-triggered protection switching and OSPF neighbor relationship flapping suppression are configured, they both take effect. Hold-down mode takes precedence over Hold-max-cost mode, followed by the configured link cost.

OSPF Flush Source Tracing

Context

If network-wide OSPF LSA flush causes network instability, source tracing must be implemented as soon as possible to locate and isolate the fault source. However, OSPF itself does not support source tracing. A conventional solution is to isolate nodes one by one until the fault source is located, but the process is complex and time-consuming and may compromise network services. To solve the preceding problem, OSPF introduces a proprietary protocol, namely, the source tracing protocol. This protocol supports the flooding of flush source information. When the preceding problem occurs, you can quickly query the flush source information on any device on the network to quickly locate the fault source.

Related Concepts

Source tracing

A mechanism that helps locate the device that flushes OSPF LSAs. This feature has the following characteristics:

- Uses a new UDP port. Source tracing packets are carried by UDP packets, and the UDP packets also carry the OSPF LSAs flushed by the current device and are flooded hop by hop based on the OSPF topology.
- Depends on OSPF neighbors for flooding and forwards packets along UDP channels, which are independent of the channels used to transmit OSPF packets. Therefore, this protocol facilitates incremental deployment. In addition, source tracing does not affect the devices with the related UDP port disabled.
- Supports query of the node that flushed LSAs on any of the devices after source tracing packets are flooded on the network, which speeds up fault locating and faulty node isolation.

Flush

Network-wide OSPF LSAs are deleted.

PS-Hello packets

Packets used to negotiate the OSPF flush source tracing capability between OSPF neighbors.

PS-LSA

When a device flushes an OSPF LSA, it generates a PS-LSA carrying information about the device and brief information about the OSPF LSA.

PS-LSU packets

OSPF flush LSA source tracing packets that carry PS-LSAs.

PS-LSU ACK packets

Acknowledgment packets used to improve OSPF flush source tracing packets.

OSPF flush source tracing port

ID of the UDP port that receives and sends OSPF flush source tracing packets. The ID is configurable.

Fundamentals

The implementation of OSPF flush source tracing is as follows:

1. Source tracing capability negotiation

After an OSPF neighbor relationship is established between two devices, they need to negotiate the source tracing capability through PS-Hello packets.

2. PS-LSA generation and flooding

When a device flushes an OSPF LSA, it generates a PS-LSA carrying information about the device and brief information about the OSPF LSA, adds the PS-LSA to a PS-LSU packet, and floods the PS-LSU packet to source tracing-capable neighbors, which helps other devices locate the fault source and perform isolation.

NOTE

Only router-LSAs, network-LSAs, and inter-area-router-LSAs can be flushed. Therefore, a device generates a PS-LSA only when it flushes a router-LSA, network-LSA, or inter-area-router-LSA.

Source tracing capability negotiation

The source tracing protocol uses UDP to carry source tracing packets and listens to the UDP port, which is used to receive and send source tracing packets. If a source tracing-capable Huawei device sends source tracing packets to a source tracing-incapable Huawei device or non-Huawei device, the source tracing-capable Huawei device may be incorrectly identified as an attacker. Therefore, the source tracing capability needs to be negotiated between the devices. In addition, the source tracing-capable device needs to send source tracing information on behalf of the source tracing-incapable device, which also requires negotiation.

Source tracing capability negotiation depends on OSPF neighbor relationships. Specifically, after an OSPF neighbor relationship is established, the local device initiates source tracing capability negotiation. [Figure 1-87](#) shows the negotiation process.

Figure 1-87 Source tracing capability negotiation

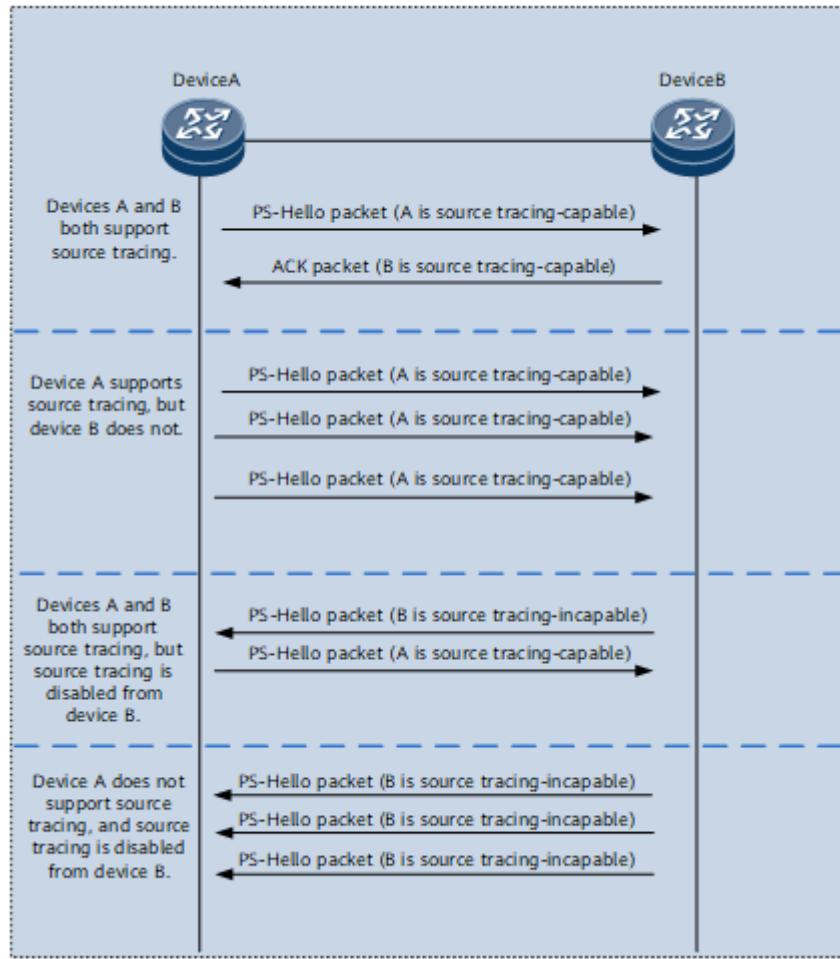


Table 1-31 Source tracing capability negotiation

Whether Source Tracing Is Supported	Source Tracing Capability Negotiation Process
Devices A and B both support source tracing.	<ol style="list-style-type: none"> 1. DeviceA sends a PS-Hello packet to notify its source tracing capability. 2. Upon reception of the PS-Hello packet, DeviceB sets the source tracing field for DeviceA and replies with an ACK packet to notify its source tracing capability to DeviceA. 3. Upon reception of the ACK packet, DeviceA sets the source tracing field for DeviceB, and does not retransmit the PS-Hello packet.

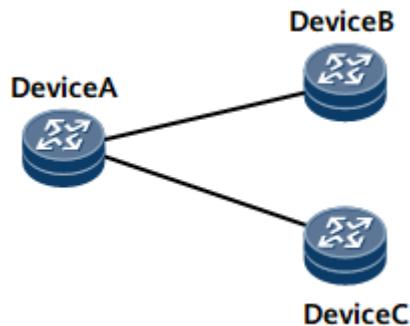
Whether Source Tracing Is Supported	Source Tracing Capability Negotiation Process
DeviceA supports source tracing, but DeviceB does not.	<ol style="list-style-type: none"> 1. DeviceA sends a PS-Hello packet to notify its source tracing capability. 2. DeviceA fails to receive an ACK packet from DeviceB after 10s elapses and retransmits the PS-Hello packet. A maximum of two retransmissions are allowed. After DeviceA fails to receive an ACK packet from DeviceB after the PS-Hello packet is retransmitted twice, DeviceA considers that DeviceB does not support source tracing.
Devices A and B both support source tracing, but source tracing is disabled from DeviceB.	<ol style="list-style-type: none"> 1. After source tracing is disabled from DeviceB, DeviceB sends a PS-Hello packet to notify its source tracing incapability. 2. Upon reception of the PS-Hello packet from DeviceB, DeviceA replies with an ACK packet that carries the source tracing capability. 3. Upon reception of the ACK packet from DeviceA, DeviceB considers the capability negotiation complete and disables the UDP port.
DeviceA does not support source tracing, and source tracing is disabled from DeviceB.	<ol style="list-style-type: none"> 1. After source tracing is disabled from DeviceB, DeviceB sends a PS-Hello packet to notify its source tracing incapability. 2. DeviceB fails to receive an ACK packet from DeviceA after 10s elapses and retransmits the PS-Hello packet. A maximum of two retransmissions are allowed. After two retransmissions, DeviceB considers the capability negotiation complete and disables the UDP port.

PS-LSA generation and flooding

PS-LSA: carries information about the node that flushed OSPF LSAs.

- If a device flushes an LSA, it generates and floods a PS-LSA to source tracing-capable neighbors.
- If a device receives a flush LSA from a source tracing-incapable neighbor, the device generates and floods a PS-LSA to source tracing-capable neighbors. If a device receives the same flush LSA (with the same LSID and sequence number) from more than one source tracing-incapable neighbor, the device generates only one PS-LSA.
- If a device flushes a router-LSA, network-LSA, or inter-area-router-LSA, it generates a PS-LSA, adds the PS-LSA to a PS-LSU packet, and floods the PS-LSU packet to all source tracing-capable neighbors.

Figure 1-88 PS-LSA generation rules



PS-LSA generation rules

- When DeviceA flushes a router-LSA, network-LSA, or inter-area-router-LSA, it generates a PS-LSA in which the **Flush Router** field is its router ID and the **Neighbor Router** field is 0, and adds the PS-LSA to the queue where packets are to be sent to all source tracing-capable neighbors.
- After DeviceA receives the flush LSA from source tracing-incapable DeviceB, DeviceA generates a PS-LSA in which the **Flush Router** field is its router ID and the **Neighbor Router** field is the router ID of DeviceB, and adds the PS-LSA to the queue where packets are to be sent to all source tracing-capable neighbors.
- After DeviceA receives the flush LSA from DeviceB, followed by the same flush LSA sent by DeviceC, DeviceA generates a PS-LSA in which the **Flush Router** field is its router ID and the **Neighbor Router** field is the router ID of DeviceB, and adds the PS-LSA to the queue where packets are to be sent to all source tracing-capable neighbors. No PS-LSA is generated in response to the flush LSA received from DeviceC.

PS-LSU packet sending rules

- During neighbor relationship establishment, a device initializes the sequence number of the PS-LSU packet of the neighbor. When the device replies with a PS-LSU packet, it adds the sequence number of the PS-LSU packet of the neighbor. During PS-LSU packet retransmission, the sequence number remains unchanged. After the device receives a PS-LSU ACK packet with the same sequence number, it increases the sequence number of the neighbor's PS-LSU packet by 1.
- The neighbor manages the PS-LSA sending queue. When a PS-LSA is added to the queue which was empty, the neighbor starts a timer. After the timer expires, the neighbor adds the PS-LSA to a PS-LSU packet, sends the packet to its neighbor, and starts another timer to wait for a PS-LSU ACK packet.
- After the PS-LSU ACK timer expires, the PS-LSU packet is retransmitted.
- When the device receives a PS-LSU ACK packet with a sequence number same as that in the neighbor record, the device clears PS-LSAs from the neighbor queue, and sends another PS-LSU packet after the timer expires.
 - If the sequence number of a received PS-LSU ACK packet is less than that in the neighbor record, the device ignores the packet.
 - If the sequence number of a received PS-LSU ACK packet is greater than that in the neighbor record, the device discards the packet.

 NOTE

PS-LSU packet sending is independent among neighbors.

PS-LSU packet receiving rules

- When a device receives a PS-LSU packet from a neighbor, the neighbor records the sequence number of the packet and replies with a PS-LSU ACK packet.
- When the device receives a PS-LSU packet with the sequence number the same as that in the neighbor record, the device discards the PS-LSU packet.
- After the device parses a PS-LSU packet, it adds the PS-LSA in the packet to its LSDB. The device also checks whether the PS-LSA is newer than the corresponding PS-LSA in its LSDB.
 - If the received PS-LSA is newer, the device floods it to other neighbors.
 - If the received PS-LSA is the same as the corresponding local one, the device does not process the received PS-LSA.
 - If the received PS-LSA is older, the device floods the corresponding local one to the neighbor.
- If the device receives a PS-LSU packet from a neighbor and the neighbor does not support source tracing, the device modifies the neighbor status as source tracing capable.

Source Tracing Security

The source tracing protocol uses a UDP port to receive and send source tracing packets. Therefore, the security of the port must be taken into consideration.

The source tracing protocol inevitably increases packet receiving and sending workload and intensifies bandwidth pressure. To minimize its impact on other protocols, the number of source tracing packets must be controlled.

The following security measures are available:

Table 1-32 Security measures for source tracing

Security Measures for Source Tracing	Fundamentals
Authentication	Source tracing is embedded in OSPF, inherits existing OSPF configuration parameters, and uses OSPF authentication parameters to authenticate packets.

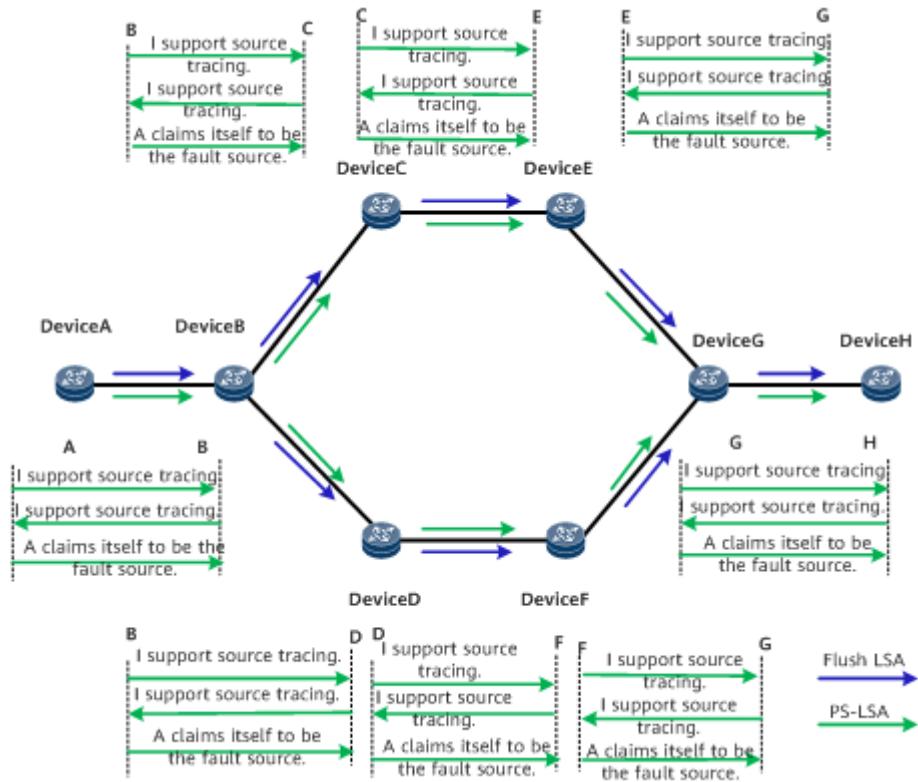
Security Measures for Source Tracing	Fundamentals
GTSM	<p>GTSM is a security mechanism that checks whether the time to live (TTL) value in each received IP packet header is within a pre-defined range.</p> <p>Source tracing packets can only be flooded as far as one hop. Therefore, GTSM can be used to check such packets by default.</p> <ul style="list-style-type: none">• When a device sends a packet, it sets the TTL of the packet to 255.• If the TTL is not 254 when the packet is received, the packet will be discarded.
CPU-CAR	<p>Interface boards can check the packets to be sent to the CPU for processing and prevent the main control board from being overloaded by a large number of packets that are sent to the CPU. The source tracing protocol needs to apply for an independent CAR channel and has small CAR values configured.</p>

Typical Scenarios

Scenario where all nodes support source tracing

All nodes on the network support source tracing, and DeviceA is the faulty source. [Figure 1-89](#) shows the networking.

Figure 1-89 Scenario where all nodes support source tracing

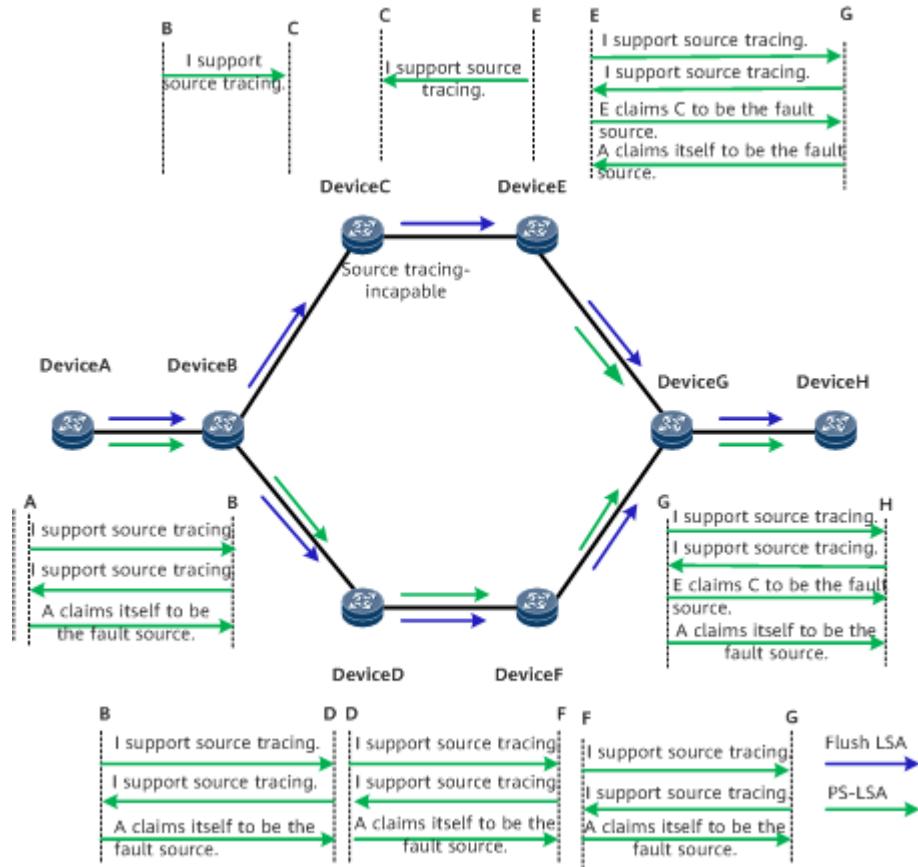


When DeviceA flushes an OSPF LSA, it generates a PS-LSA that carries DeviceA information and brief information about the flush LSA and floods it. After the fault occurs, maintenance personnel can log in to any node on the network to locate DeviceA, which keeps sending flush LSAs, and isolate DeviceA from the network.

Scenario where source tracing-incapable nodes are not isolated from source tracing-capable nodes

All nodes on the network except DeviceC support source tracing, and DeviceA is the fault source. In this case, the PS-LSA can be flooded on the entire network, and the fault source can be accurately located. [Figure 1-90](#) shows the networking.

Figure 1-90 Scenario where source tracing-incapable nodes are not isolated from source tracing-capable nodes



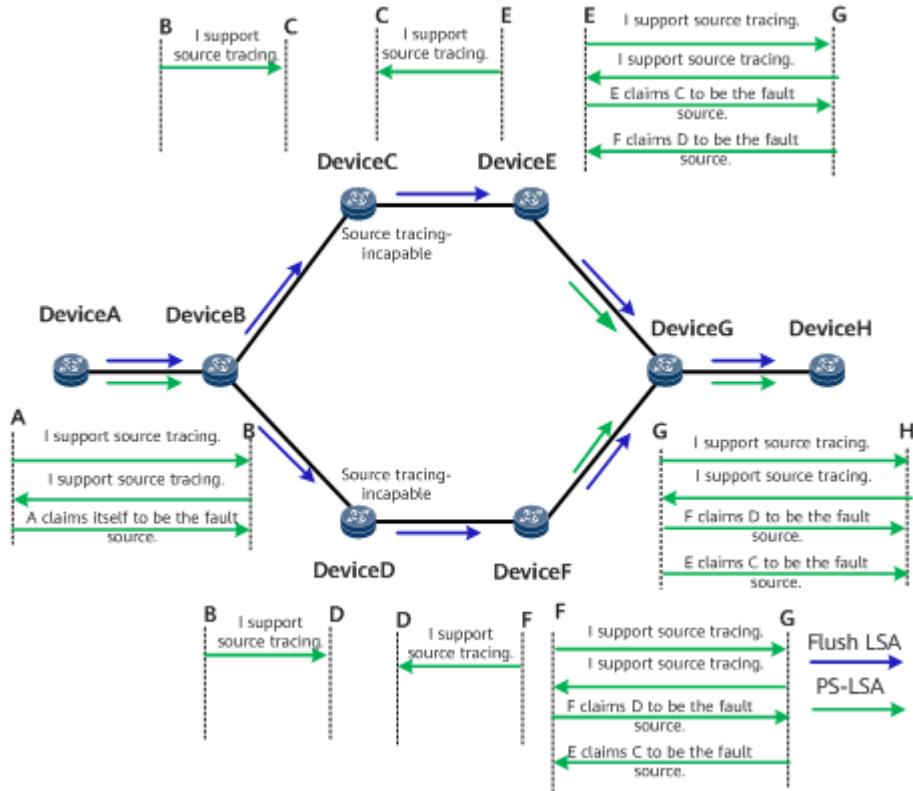
When DeviceA flushes an OSPF LSA, it generates a PS-LSA that carries DeviceA information and brief information about the flush LSA and floods it. When DeviceB and DeviceE negotiate the source tracing capability with DeviceC, they find that DeviceC does not support source tracing. Therefore, after DeviceB receives the PS-LSA from DeviceA, DeviceB sends the PS-LSA to DeviceD, but not to DeviceC. After receiving the flush LSA from DeviceC, DeviceE generates a PS-LSA that carries information about the advertisement source (DeviceE), flush source (DeviceC), and the OSPF flush LSA, and floods the PS-LSA on the network.

After the fault occurs, maintenance personnel can log in to any device on the network except DeviceC to locate the faulty node. Two possible faulty nodes can be located in this case: DeviceA and DeviceC, and they both send the same flush LSA. In this case, DeviceA takes precedence over DeviceC when the maintenance personnel determine the most possible faulty source. After DeviceA is isolated, the network recovers.

Scenario where source tracing-incapable nodes are isolated from source tracing-capable nodes

All nodes on the network except DeviceC and DeviceD support source tracing, and DeviceA is the fault source. In this case, the PS-LSA cannot be flooded on the entire network. [Figure 1-91](#) shows the networking.

Figure 1-91 Scenario where source tracing-incapable nodes are isolated from source tracing-capable nodes



When DeviceA flushes an OSPF LSA, it generates a PS-LSA that carries DeviceA information and brief information about the flush LSA and floods it. However, the PS-LSA sent by DeviceA can reach only DeviceB because DeviceC and DeviceD do not support source tracing.

During source tracing capability negotiation, DeviceE finds that DeviceC does not support source tracing, and DeviceF finds that DeviceD does not support source tracing. After DeviceE receives the flush LSA from DeviceC, DeviceE generates and floods a PS-LSA on behalf of DeviceC. Similarly, after DeviceF receives the flush LSA from DeviceD, DeviceF generates and floods a PS-LSA on behalf of DeviceD.

After the fault occurs:

- If maintenance personnel log in to DeviceA or DeviceB, the personnel can locate the fault source (DeviceA) directly. After DeviceA is isolated, the network recovers.
- If the maintenance personnel log in to DeviceE, DeviceF, DeviceG, or DeviceH, the personnel will find that DeviceE claims DeviceC to be the fault source of the OSPF flush LSA and DeviceF claims DeviceD to be the fault source of the same OSPF flush LSA.
- If the maintenance personnel log in to DeviceC and DeviceD, the personnel will find that the flush LSA was initiated by DeviceB, not generated by DeviceC or DeviceD.
- If the maintenance personnel log in to DeviceB, the personnel will find that DeviceA is the faulty device, and isolate DeviceA. After DeviceA is isolated, the network recovers.

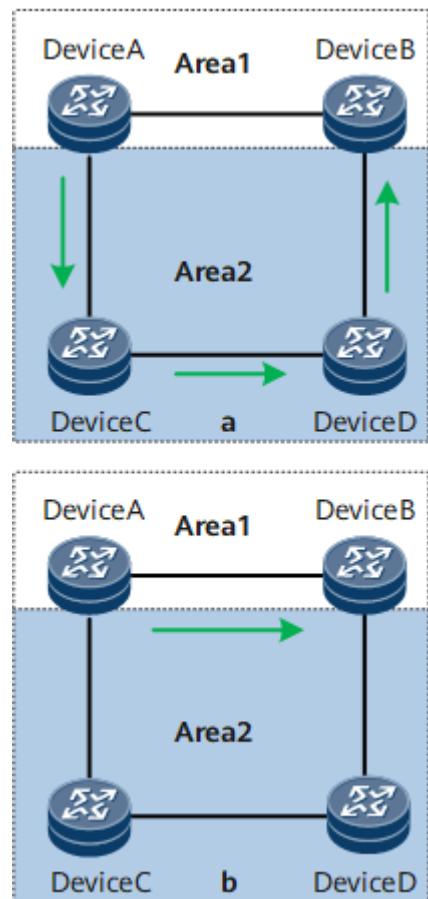
OSPF Multi-Area Adjacency

Context

OSPF prefers the shortest path during intra-area route calculation. Even if an inter-area path is the shortest, OSPF still prefers an intra-area route. Each OSPF interface belongs to only one area. As a result, even if a high-speed link exists in an area, traffic of another area cannot be forwarded along this link. The preceding problem can be solved by configuring multiple sub-interfaces and adding them to different areas. However, the disadvantage of this method is that multiple sub-interfaces need to be enabled and each sub-interface needs to be configured with an independent IP address. As a result, a large number of IP addresses are advertised, increasing the total number of routes. To address this issue, OSPF multi-area adjacency is introduced.

OSPF multi-area adjacency allows an OSPF interface to be added to multiple areas so that a link can be shared by these areas.

Figure 1-92 Traffic forwarding paths before and after OSPF multi-area adjacency is configured



On the network shown in [Figure 1-92](#), the link between DeviceA and DeviceB is a high-speed link in area 1.

Before OSPF multi-area adjacency is configured, traffic from DeviceA to DeviceB in area 2 is forwarded along a low-speed path (DeviceA -> DeviceC -> DeviceD -> DeviceB), as shown in [Figure 1-92 a](#).

After OSPF multi-area adjacency interfaces are configured on DeviceA and DeviceB and are added to area 2, traffic from DeviceA to DeviceB in area 2 is forwarded along the high-speed link (DeviceA -> DeviceB), as shown in [Figure 1-92 b](#).

This feature has the following advantages:

- Allows interface multiplexing, thereby reducing the consumption of OSPF interface resources in multi-area scenarios.
- Allows link multiplexing, which prevents a traffic detour to low-speed links and optimizes the OSPF network.

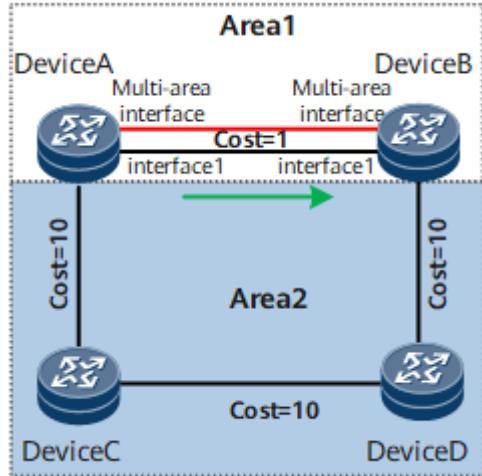
Related Concepts

Multi-area adjacency interface: OSPF logical interface created by enabling multi-area adjacency on an OSPF primary interface. A multi-area adjacency interface is also called a secondary interface. The multi-area adjacency interface has the following characteristics:

- The multi-area adjacency interface and the primary interface belong to different OSPF areas.
- The network type of the multi-area adjacency interface can only be P2P. The multi-area adjacency interface runs an independent interface state machine and neighbor state machine.
- The multi-area adjacency interface and primary interface share the same interface index and packet transmission channel. Whether the multi-area adjacency interface or the primary interface is selected to forward an OSPF packet is determined by the area ID carried in the packet header and related configuration.
- The multi-area adjacency interface of a P2P interface sends protocol packets in multicast mode.
- The multi-area adjacency interface of a non-P2P interface sends protocol packets in unicast mode.

Fundamentals

Figure 1-93 Networking for OSPF multi-area adjacency



On the network shown in [Figure 1-93](#), the link between DeviceA and DeviceB is a high-speed link in area 1. In area 2, traffic from DeviceA to DeviceB is forwarded along a low-speed path (DeviceA -> DeviceC -> DeviceD -> DeviceB), as shown in [Figure 1-93](#). To forward traffic from DeviceA to DeviceB in area 2 along the high-speed link DeviceA -> DeviceB, configure OSPF multi-area adjacency.

Specifically, create OSPF multi-area adjacency interfaces on the primary interfaces of DeviceA and DeviceB and add the multi-area adjacency interfaces to area 2.

1. An adjacency is established between the multi-area adjacency interfaces of DeviceA and DeviceB. The process of establishing such an adjacency is the same as that of establishing a common OSPF adjacency. For details, see [Adjacency Establishment](#).
2. OSPF performs route calculation. The calculation process is the same as that in the basic OSPF protocol. For details, see [Route Calculation](#).

The optimal path calculated by OSPF in area 2 is the path DeviceA -> DeviceB. Therefore, traffic from DeviceA to DeviceB is forwarded along the high-speed link DeviceA -> DeviceB. In this manner, this high-speed link can be shared by area 1 and area 2.

OSPF NSR

Non-Stop Routing (NSR) is a routing technique that prevents neighboring routers from detecting faults on the control plane of a device with a backup control plane. With NSR, when the control plane of the device fails, the neighbor relationship set up through specific routing protocols, MPLS, and other protocols that carry services is uninterrupted.

As networks develop and expand, carriers place increasingly higher requirements on the reliability of IP networks. NSR was introduced as a high availability (HA) solution to ensure that services transmitted by a device were unaffected by hardware or software failures on the device.

OSPF NSR synchronizes the protocol data on the master main control board to the slave main control board in real time. When the master main control board fails or

needs to be upgraded, the slave main control board immediately takes over services from the master main control board, and the neighbor remains unaware the fault occurred. Real-time data synchronization between the master and slave main control boards through OSPF NSR is implemented as follows:

- OSPF backs up configuration data and dynamic data, including information about interfaces, neighbors, and LSDBs.
- OSPF does not back up routes, shortest path trees (SPTs), or Traffic Engineering Databases (TEDBs). Such information can be restored using the source data in the database backup process.
- When the master/slave main control board switchover occurs, the new master main control board restores the operations data and takes over services from the former master main control board. Throughout this process, the neighbor remains unaware of the fault.

OSPF IP FRR

OSPF IP fast reroute (FRR) refers to the process by which OSPF precomputes a backup path based on the network-wide LSDBs, and stores this backup path in the forwarding table. If the primary path fails, traffic can be quickly switched to the backup path.

Context

As networks develop, voice over IP (VoIP) and online video services pose higher requirements for real-time transmission. Nevertheless, if a primary link fails, OSPF-enabled devices need to perform multiple operations, including detecting the fault, updating the link-state advertisement (LSA), flooding the LSA, calculating routes, and delivering forward information base (FIB) entries before switching traffic to a new link. This process takes a much longer time, the minimum delay to which users are sensitive. As a result, the requirements for real-time transmission cannot be met. OSPF IP FRR can solve this problem. OSPF IP FRR conforms to dynamic IP FRR defined by standard protocols. With OSPF IP FRR, devices can switch traffic from a faulty primary link to a backup link, protecting against a link or node failure.

Major FRR techniques include loop-free alternate (LFA), U-turn, Not-Via, Remote LFA, and MRT, among which OSPF supports only LFA and Remote LFA.

Related Concepts

OSPF IP FRR

OSPF IP FRR refers to a mechanism in which a device uses the loop-free alternate (LFA) algorithm to compute the next hop of a backup link and stores the next hop together with the primary link in the forwarding table. If the primary link fails, the device switches the traffic to the backup link before routes are converged on the control plane. This mechanism keeps the traffic interruption duration and minimizes the impacts.

OSPF IP FRR policy

An OSPF IP FRR policy can be configured to filter alternate next hops. Only the alternate next hops that match the filtering rules of the policy can be added to the IP routing table.

LFA algorithm

A device uses the shortest path first (SPF) algorithm to calculate the shortest path from each neighbor with a backup link to the destination node. The device then uses the inequalities defined in standard protocols and the LFA algorithm to calculate the next hop of the loop-free backup link that has the smallest cost of the available shortest paths.

Remote LFA

LFA FRR cannot be used to calculate alternate links on large-scale networks, especially on ring networks. Remote LFA FRR addresses this problem by calculating a PQ node and establishing a tunnel between the source node of a primary link and the PQ node. If the primary link fails, traffic can be automatically switched to the tunnel, which improves network reliability.

P space

Remote LFA uses the source end of the primary link as the root node and calculates an SPT to all the other nodes on the network (with the primary link calculated in the tree). After all the nodes along the primary link are removed from the SPT, the remaining nodes on the SPT form the P space.

Extended P space

Remote LFA uses all neighbors of the primary link's source end as root nodes and calculates separate SPTs (with the primary link calculated in the trees). After all the nodes along the primary link are removed from each SPT, the remaining nodes on the SPTs form the extended P space.

Q space

Remote LFA uses the destination node as the root node and calculates an SPT to all the other nodes on the network (with the primary link calculated in the tree). After all the nodes along the primary link are removed from the SPT, the remaining nodes on the SPT form the Q space.

PQ node

A PQ node exists both in the extended P space and Q space and is used by Remote LFA as the destination of a protection tunnel.

OSPF LFA FRR

OSPF LFA FRR protects traffic against either a link failure or a node-and-link failure. The node-and-link protection takes precedence over the link protection.

Link protection

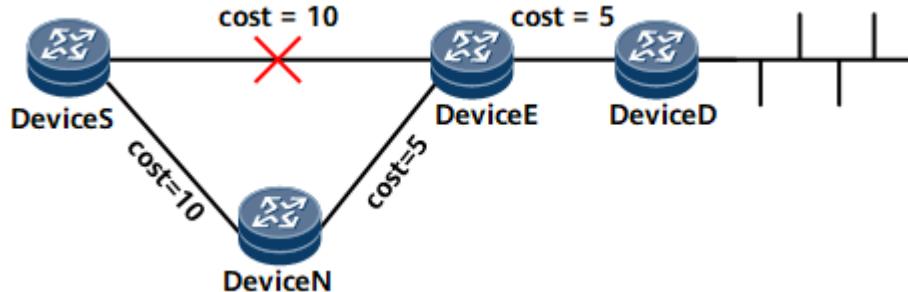
Link protection takes effect when the traffic to be protected flows along a specified link.

In [Figure 1-94](#), traffic flows from Device S to Device D. The primary link is Device S->Device E->Device D, and the backup link is Device S->Device N->Device E->Device D. If link costs meet the inequality: $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, S) + \text{Distance}_{\text{opt}}(S, D)$ and OSPF IP FRR is enabled, Device S switches the traffic to the backup link if the primary link fails, reducing the traffic interruption duration.

 NOTE

Distance_opt(X, Y) indicates the shortest link from X to Y. S stands for a source node, E for the faulty node, N for a node along a backup link, and D for a destination node.

Figure 1-94 OSPF IP FRR link protection



Node-and-link protection

Node-and-link protection takes effect when the traffic to be protected.

In [Figure 1-95](#), traffic flows from Device S to Device D. The primary link is Device S->Device E->Device D, and the backup link is Device S->Device N->Device D. The preceding inequalities are met. With OSPF IP FRR, Device S switches the traffic to the backup link if the primary link fails, reducing the traffic interruption duration.

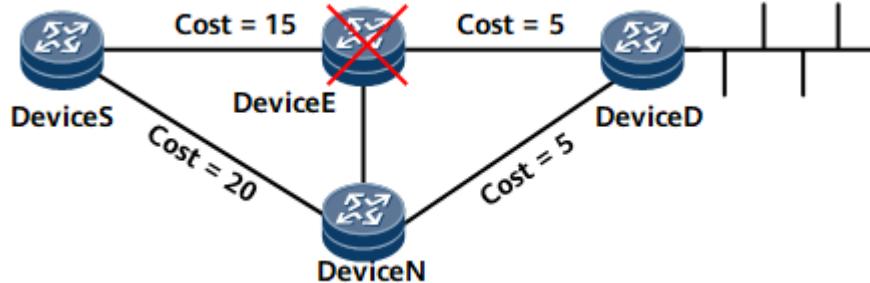
Node-and-link protection takes effect when the following conditions are met:

- The link costs meet the inequality: $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, S) + \text{Distance}_{\text{opt}}(S, D)$.
- The interface costs meet the inequality: $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, E) + \text{Distance}_{\text{opt}}(E, D)$.

 NOTE

Distance_opt(X, Y) indicates the shortest link from X to Y. S stands for a source node, E for the faulty node, N for a node along a backup link, and D for a destination node.

Figure 1-95 OSPF IP FRR node-and-link protection

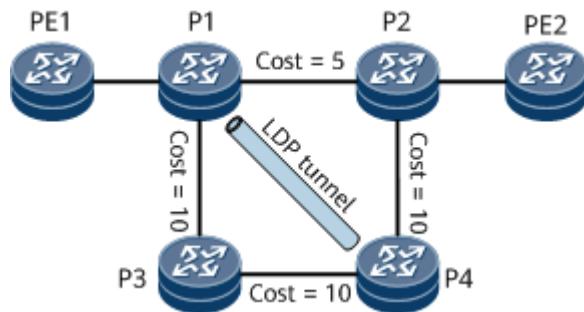


OSPF Remote LFA FRR

Similar to OSPF LFA FRR, remote LFA is also classified as link protection or node-and-link protection. The following example shows how remote LFA works to protect against link failures:

In [Figure 1-96](#), traffic flows through PE1 -> P1 -> P2 -> PE2, and the primary link is between P1 and P2. Remote LFA calculates a PQ node (P4) and establishes an LDP between P1 and P4. If P1 detects a failure on the primary link, P1 encapsulates packets into MPLS packets and forwards MPLS packets to P4. After receiving the packets, P4 removes the MPLS label from them and searches the IP routing table for a next hop to forward the packets to PE2. Remote LFA ensures uninterrupted traffic forwarding.

Figure 1-96 Networking for Remote LFA



On the network shown in [Figure 1-96](#), Remote LFA calculates the PQ node as follows:

1. SPTs are calculated, with each of P1's neighbors (excluding the neighbors on the primary link) as the root. In this case, the neighbors PE1 and P3 are used as roots. For each SPT, an extended P space is composed of the root node and those nodes reachable through non-P1-P2 links. When PE1 is used as a root node for calculation, the extended P space {PE1, P1, P3} is obtained. When P3 is used as a root node for calculation, the extended P space {PE1, P1, P3, P4} is obtained. By combining these two extended P spaces, the final extended P space {PE1, P1, P3, P4} is obtained.
2. A reverse SPT with PE2 as the root is calculated. The Q space is {P2, PE2, P4}.
3. The node that exists both in the extended P space and Q space is the PQ node. In this example, P4 is the PQ node.

NOTE

On a network with a large number of nodes, to ensure that RLFA/TI-LFA calculation can be completed as soon as possible, the elected P and Q nodes may not be optimal, but they comply with rules. This does not affect the protection effect.

OSPF microloop avoidance

In [Figure 1-96](#), OSPF remote LFA FRR is enabled, the primary link is PE1 -> P1 -> P2 -> PE2, and the backup link is PE1 -> P1 -> P3 -> P4 -> P2 -> PE2, and the link P1 -> P3 -> P4 is an LDP tunnel. If the primary link fails, traffic is switched to the backup link, and then another round of the new primary link calculation begins. Specifically, after P1 completes route convergence, its next hop becomes P3. However, the route convergence on P3 is slower than that on P1, and P3's next

hop is still P1. As a result, a temporary loop occurs between P1 and P3. OSPF microloop avoidance can address this problem by delaying P1 from switching its next hop until the next hop of P3 becomes P4. Then traffic is switched to the new primary link (PE1 -> P1 -> P3 -> P4 -> P2 -> PE2), and on the link P1 -> P3 -> P4, traffic is forwarded based on IP routes.

 NOTE

OSPF microloop avoidance applies only to OSPF remote LFA FRR.

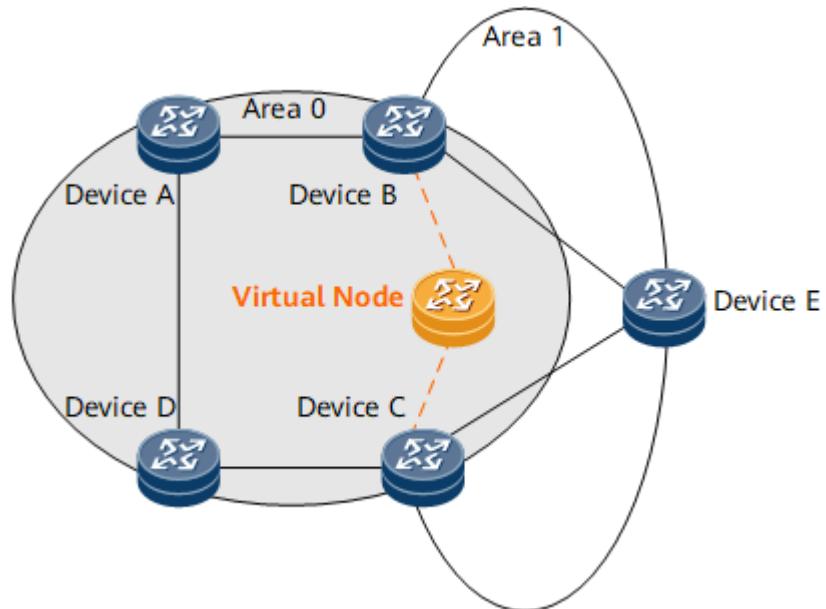
OSPF FRR in the Scenario Where Multiple Nodes Advertise the Same Route

The SPF algorithm is used by OSPF LFA FRR and OSPF Remote LFA FRR to calculate the shortest path from each neighbor (root node) that provides a backup link to the destination node. The calculation result is a node-based backup next hop. This applies to the scenarios where a route is advertised by a single node. As networks are increasingly diversified, two ABRs or ASBRs are deployed to improve network reliability. In this case, OSPF FRR in a scenario where multiple nodes advertise the same route is introduced.

 NOTE

In a scenario where multiple nodes advertise the same route, OSPF FRR is implemented by calculating the Type 3 LSAs advertised by ABRs of an area for intra-area, inter-area, ASE, or NSSA routes. Therefore, the OSPF FRR calculation methods are the same when multiple nodes advertise the same route. Inter-area routing is used as an example to describe how FRR in a multi-node routing scenario works.

Figure 1-97 OSPF FRR in the scenario where multiple nodes advertise the same route



In **Figure 1-97**, Device B and Device C function as ABRs to forward routes between area 0 and area 1. Device E advertises an intra-area route. Upon receipt of the route, Device B and Device C translate it into a Type 3 LSA and flood the LSA to area 0. After OSPF FRR is enabled on Device A, Device A considers both Device B and Device C as its neighbors. Without a fixed neighbor as the root node, Device A

fails to calculate the FRR backup next hop. To address this problem, a virtual node is simulated between Device B and Device C, converting the scenario where multiple nodes advertise the same route into the scenario where the route is received from only one node. Then the backup next hop of the virtual node is calculated based on LFA and Remote LFA. Then the route advertised by multiple nodes inherits the backup next hop from the virtual node.

For example, Device B and Device C advertise a route with prefix 10.1.1.0/24, and OSPF FRR is enabled on Device A. Device A cannot calculate a backup next hop for the route 10.1.1.0/24 because this route has two sources (Device B and Device C). To address this problem, a virtual node is simulated between Device B and Device C based on the two sources of the route 10.1.1.0/24. The virtual node forms a link with each of Device B and Device C. If the virtual node advertises a 10.1.1.0/24 route, it will use the smaller cost of the routes advertised by Device B and Device C as the cost of the route. If the cost of the route advertised by Device B is 5 and that of the route advertised by Device C is 10, the cost of the route advertised by the virtual node is 5. The cost of the link from Device B to the virtual node is 0, and that of the link from Device C to the virtual node is 5. The costs of the links from the virtual node to Device B and Device C are both 65535, the maximum value. Device A is configured to consider Device B and Device C as invalid sources of the 10.1.1.0/24 route and use the LFA or remote LFA algorithm to calculate the backup next hop for the route, with the virtual node as the root node.

In a scenario where multiple nodes advertise the same route, OSPF FRR can use the LFA or remote LFA algorithm. When OSPF FRR uses the remote LFA algorithm, PQ node selection has the following restrictions:

- An LDP tunnel will be established between a faulty node and a PQ node, and a virtual node in the scenario where multiple nodes advertise the same route cannot transmit traffic through LDP tunnels. As a result, the virtual node cannot be selected as a PQ node.
- The destination node of Remote LFA is not selected as a PQ node. After a virtual node is added in a scenario where multiple nodes advertise the same route, the destination node becomes a virtual node. Therefore, a node directly connected to the virtual node cannot be selected as a PQ node.

OSPF SRLG FRR

A shared risk link group (SRLG) is a set of links that share a common physical resource (such as a fiber). These links share the same risk level. If one of the links fails, all the other links in the SRLG may also fail.

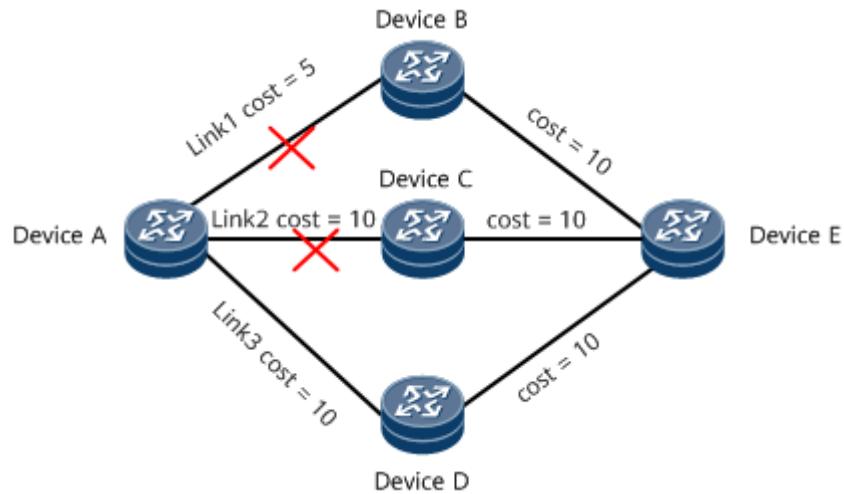
On the network shown in [Figure 1-98](#), traffic is forwarded from Device A to Device E. There are three links between Device A and Device E: Link1, Link2, and Link3. The cost of Link1 is the smallest, and the costs of Link2 and Link3 are the same. Therefore, Link1 is the primary link for traffic forwarding.

OSPF LFA IP FRR provides protection for Link1, and Link2 or Link3 is selected as the backup link for traffic forwarding. Assume that Link2 is selected as the backup link:

- If link 1 fails but the backup link (link 2) is normal, traffic can be forwarded normally after being switched to the backup link.
- If both link 1 and link 2 fail, traffic is interrupted after being switched to the backup link.

OSPF SRLG FRR can be configured in the scenario where some links have the same risk of failure. If Link1 and Link2 have the same risk of failure, you can add them to an SRLG and configure OSPF SRLG FRR so that a link outside the SRLG is preferentially selected as a backup link, which reduces the possibility of service interruptions. After Link1 and Link2 are added to the same SRLG, OSPF LFA IP FRR selects Link3, which is not in the SRLG, as the backup link to provide protection for Link1. If both Link1 and Link2 fail, traffic can be switched to Link3 for normal transmission.

Figure 1-98 Networking diagram of OSPF SRLG FRR



Derivative Functions

If you bind a Bidirectional Forwarding Detection (BFD) session with OSPF IP FRR, the BFD session goes down if BFD detects a link fault. If the BFD session goes down, OSPF IP FRR is triggered to switch traffic from the faulty link to the backup link, which minimizes the loss of traffic.

OSPF Authentication

OSPF authentication encrypts OSPF packets by adding the authentication field to packets to ensure network security. When receiving an OSPF packet from a remote device, the local device discards the packet if the authentication password carried in the packet is different from the local one, protecting the local device against potential attacks.

In terms of the packet type, the authentication is classified as follows:

- Area authentication: configured in the OSPF area view and applies to packets on all interfaces in the OSPF area.
- Interface authentication: configured in the interface view and applies to all packets on the interface.

In terms of packet the authentication modes, the authentication is classified as follows:

- Non-authentication: Authentication is not performed.
- Simple authentication: A configured password is directly added to packets for authentication. This authentication mode is insecure.

- Message-digest algorithm 5 (MD5) authentication: A configured password is hashed using an algorithm such as MD5, and the ciphertext password is added to packets for authentication. This authentication mode improves password security. Currently, MD5 and hash-based message authentication code for MD5 (HMAC-MD5) are supported. For the sake of security, using the HMAC-SHA256 algorithm rather than the MD5 algorithm is recommended.
- Keychain authentication: A keychain consists of multiple authentication keys, each of which contains an ID and a password. Each key in a keychain has a lifecycle, based on which keys are dynamically selected. A keychain can also dynamically select an authentication key to enhance attack defense.
Keychain improves OSPF security by dynamically changing algorithms and keys. Keychain authentication can be used to authenticate OSPF packets and the process of establishing a TCP connection as well.
- HMAC-SHA256 authentication: A configured password is hashed using the HMAC for secure hash algorithm 256 (HMAC-SHA256) algorithm, and the ciphertext password is added to packets for authentication. This authentication mode improves password security.

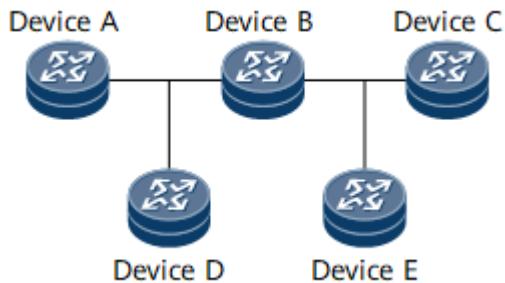
OSPF carries authentication types in packet headers and authentication information in packet tails.

The authentication types include:

- 0: non-authentication
- 1: simple authentication
- 2: Ciphertext authentication

Usage Scenario

Figure 1-99 OSPF authentication on a broadcast network



The configuration requirements are as follows:

- The interface authentication configurations must be the same on all devices on the same network so that OSPF neighbor relationships can be established.
- The area authentication configurations must be the same on all devices in the same area.

OSPF Packet Format

The OSPF protocol number is 89. OSPF packets are encapsulated into IP packets. OSPF packets are classified into five types of packets: Hello packets, DD packets, LSR packets, LSU packets, and LSAck packets.

- [Hello packet](#)
- [DD packet](#)
- [LSR packet](#)
- [LSU packet](#)
- [LSAck packet](#)

Packet Header Format

The five types of OSPF packets have the same packet header format. The length of an OSPF packet header is 24 bytes. [Figure 1-100](#) shows an OSPF packet header.

Figure 1-100 Packet header format

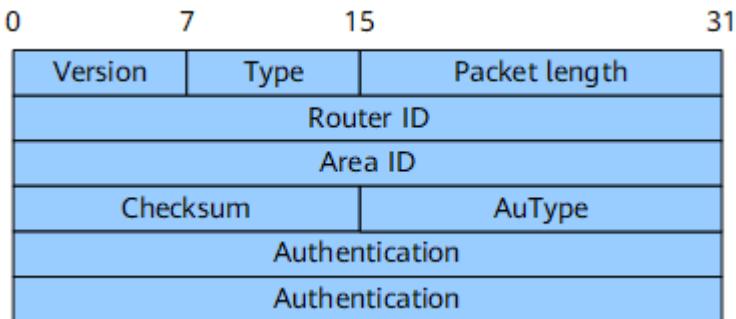


Table 1-33 OSPF packet header fields

Field	Length	Description
Version	8 bits	OSPF version number. For OSPFv2, the value is 2.
Type	8 bits	OSPF packet type. The values are as follows: <ul style="list-style-type: none">● 1: Hello packet● 2: DD packet● 3: LSR packet● 4: LSU packet● 5: LSAck packet
Packet length	16 bits	Length of the OSPF packet with the packet header, in bytes.
Router ID	32 bits	ID of the router that sends the OSPF packet.
Area ID	32 bits	ID of the area to which the router that sends the OSPF packet belongs.
Checksum	16 bits	Checksum of the OSPF packet that does not carry the Authentication field.

Field	Length	Description
AuType	16 bits	<p>Authentication type. The values are as follows:</p> <ul style="list-style-type: none">• 0: non-authentication• 1: simple authentication• 2: message digest algorithm 5 (MD5) authentication <p>NOTE The MD5 algorithm is insecure and poses security risks.</p>
Authentication	64 bits	This field has different meanings for different AuType values: <ul style="list-style-type: none">• 0: This field is not defined.• 1: This field defines password information.• 2: This field contains the key ID, MD5 authentication data length, and sequence number.

 **NOTE**

MD5 authentication data is added to an OSPF packet and is not included in the Authentication field.

Hello Packet

Hello packets are commonly used packets, which are periodically sent by OSPF interfaces to establish and maintain neighbor relationships. A Hello packet includes information about the designated router (DR), backup designated router (BDR), timers, and known neighbors. [Figure 1-101](#) shows the format of a Hello packet.

Figure 1-101 Format of a Hello packet

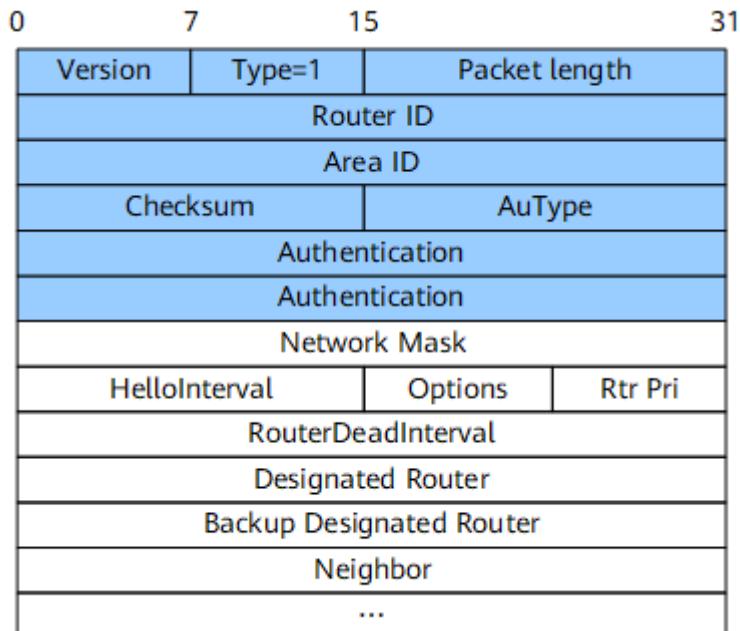


Table 1-34 Hello packet fields

Field	Length	Description
Network Mask	32 bits	Mask of the network on which the interface that sends the Hello packet resides.
HelloInterval	16 bits	Interval at which Hello packets are sent.
Options	8 bits	The values are as follows: <ul style="list-style-type: none"> • E: AS-external-LSAs can be flooded. • MC: IP multicast packets are forwarded. • N/P: Type 7 LSAs are processed. • DC: On-demand links are processed.
Rtr Pri	8 bits	DR priority. The default value is 1. NOTE If the DR priority of a router interface is set to 0, the interface cannot participate in a DR or BDR election.
RouterDeadInterval	32 bits	Dead interval. If a device does not receive any Hello packets from its neighbors within a specified dead interval, the neighbors are considered down.
Designated Router	32 bits	Interface address of the DR.
Backup Designated Router	32 bits	Interface address of the BDR.
Neighbor	32 bits	Router ID of the neighbor.

Table 1-35 lists the address types, interval types, and default intervals used when Hello packets are transmitted on different networks.

Table 1-35 Hello packet characteristics for various network types

Network Type	Address Type	Interval Type	Default Interval
Broadcast	Multicast address	HelloInterval	10 seconds
NBMA	Unicast address	<ul style="list-style-type: none"> • HelloInterval is used by the DR, BDR, and router that can become a DR. • PollInterval is used when neighbors become Down, and HelloInterval is used in other cases. 	30 seconds for HelloInterval 120 seconds for PollInterval

Network Type	Address Type	Interval Type	Default Interval
P2P	Multicast address	HelloInterval	10 seconds
P2MP	Multicast address	HelloInterval	30 seconds

NOTE

Routers on the same network segment must have the same HelloInterval and RouterDeadInterval values. Otherwise, they cannot establish neighbor relationships. In addition, on an NBMA network, the PollInterval values must be the same at both ends.

DD Packet

During an adjacency initialization, two routers use DD packets to describe their own link state databases (LSDBs) for LSDB synchronization. A DD packet contains the header of each LSA in an LSDB. An LSA header uniquely identifies an LSA. The LSA header occupies only a small portion of the LSA, which reduces the amount of traffic transmitted between routers. A neighbor can use the LSA header to check whether it already has the LSA. When two routers exchange DD packets, one functions as the master, and the other functions as the slave. The master defines a start sequence number and increases the sequence number by one each time it sends a DD packet. After the slave receives a DD packet, it uses the sequence number carried in the DD packet for acknowledgment.

[Figure 1-102](#) shows the format of a DD packet.

[Figure 1-102](#) Format of a DD packet

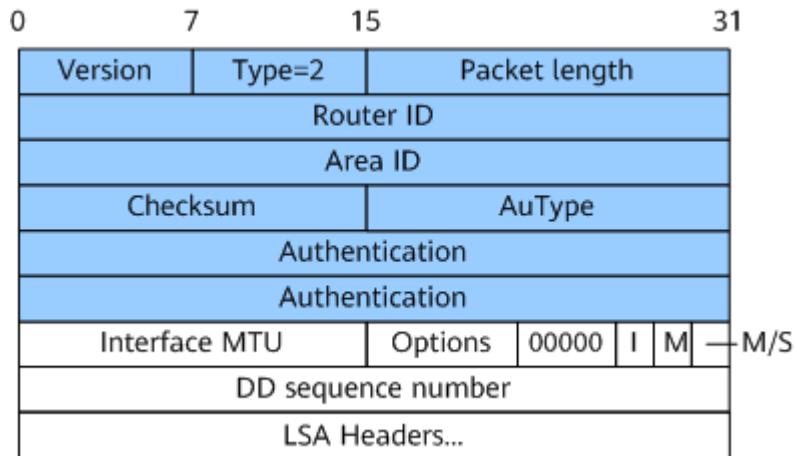


Table 1-36 DD packet fields

Field	Length	Description
Interface MTU	16 bits	Maximum size of an IP packet that an interface can send without fragmenting the packet.
Options	8 bits	The values are as follows: <ul style="list-style-type: none">• E: AS-external-LSAs can be flooded.• MC: IP multicast packets are forwarded.• N/P: Type 7 LSAs are processed.• DC: On-demand links are processed.
I	1 bit	If the DD packet is the first among multiple consecutive DD packets sent by a device, this field is set to 1. Otherwise, this field is set to 0.
M (More)	1 bit	If the DD packet is the last among multiple consecutive DD packets sent by a device, this field is set to 0. Otherwise, this field is set to 1.
M/S (Master/Slave)	1 bit	When two OSPF devices exchange DD packets, they negotiate a master/slave relationship. The device with a larger router ID becomes the master. If this field is set to 1, the device that sends the DD packet is the master.
DD sequence number	32 bits	Sequence number of the DD packet. The master and slave use the sequence number to ensure that DD packets are correctly transmitted.
LSA Headers	-	LSA header information included in the DD packet.

LSR Packet

After two routers exchange DD packets, they send LSR packets to request LSAs from each other. The LSR packets contain the summaries of the requested LSAs. [Figure 1-103](#) shows the format of an LSR packet.

Figure 1-103 Format of an LSR packet

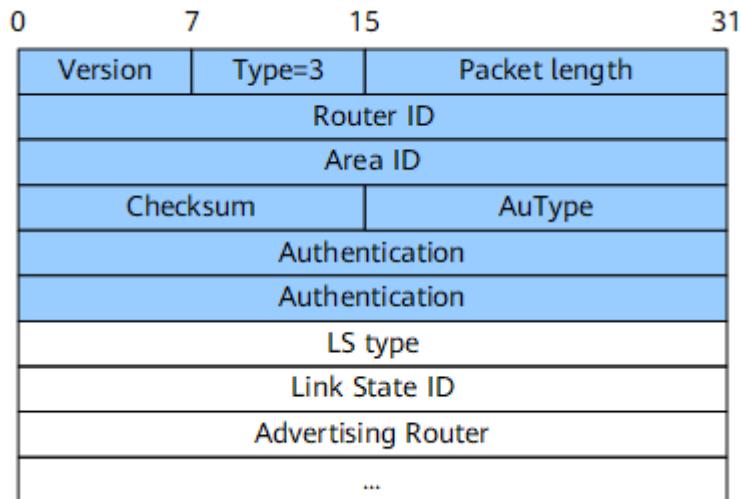


Table 1-37 LSR packet fields

Field	Length	Description
LS type	32 bits	Type of the LSA.
Link State ID	32 bits	This field together with the LS type field describes an LSA in an AS.
Advertising Router	32 bits	Router ID of the router that generates the LSA.

NOTE

The LS type, Link State ID, and Advertising Router fields can uniquely identify an LSA. If the preceding fields of two LSAs are the same, the device uses the LS sequence number, LS checksum, and LS age fields to determine which LSA is newer.

LSU Packet

A router uses an LSU packet to transmit LSAs requested by its neighbors or to flood its own updated LSAs. The LSU packet contains a set of LSAs. On networks that support multicast and broadcast, LSU packets are multicast to flood LSAs. To ensure reliable LSA flooding, a device uses an LSAck packet to acknowledge the LSAs contained in an LSU packet that is received from a neighbor. If an LSA fails to be acknowledged, the LSA is directly retransmitted to the neighbor. [Figure 1-104](#) shows the format of an LSU packet.

Figure 1-104 Format of an LSU packet

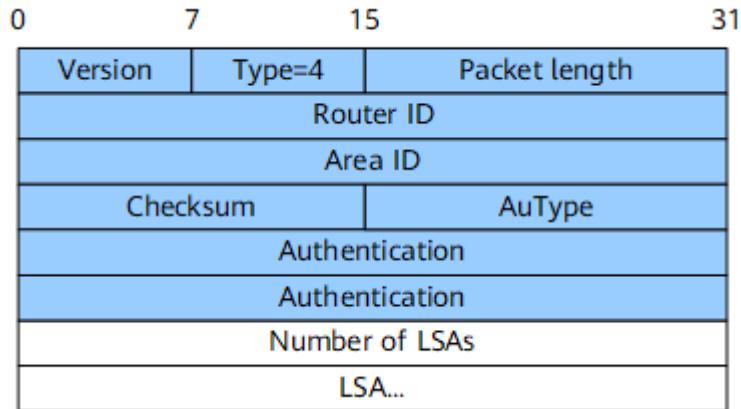


Table 1-38 LSU packet field

Field	Length	Description
Number of LSAs	32 bits	Number of LSAs contained in the LSU packet

LSAck Packet

A device uses an LSAck packet to acknowledge the headers of the LSAs contained in a received LSU packet. LSAck packets are transmitted in unicast or multicast mode according to the link type. [Figure 1-105](#) shows the format of an LSAck packet.

Figure 1-105 Format of an LSAck packet

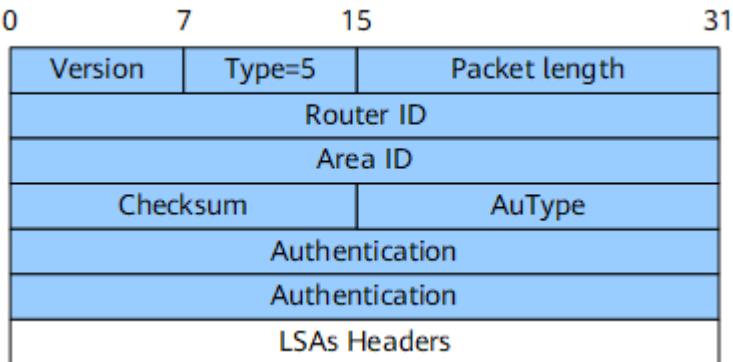


Table 1-39 LSAck packet field

Field	Length	Description
LSAs Headers	Determined by the header length of the LSA to be acknowledged.	This field is used to acknowledge an LSA.

OSPF LSA Format

Each router in an AS generates one or more types of LSAs based on its router type. A set of LSAs forms a link state database (LSDB). In OSPF, routing information is encapsulated in LSAs for advertisement. Common LSAs are as follows:

- **Router-LSAs**
- **Network-LSAs**
- **Summary-LSAs** (including Network-Summary-LSAs and ASBR-Summary-LSAs)
- **AS-External-LSAs**

LSA Header Information

All LSAs have the same header format, as shown in [Figure 1-106](#).

Figure 1-106 LSA header format

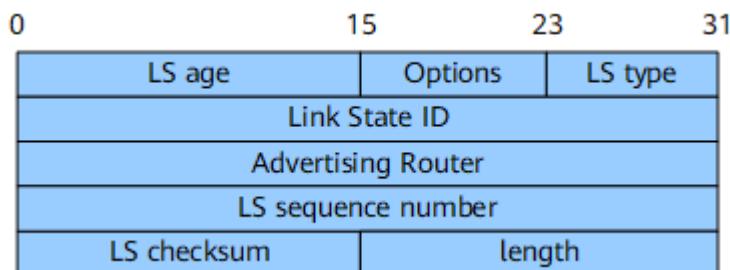


Table 1-40 LSA header fields

Field	Length	Description
LS age	16 bits	Time that has elapsed since an LSA is generated, in seconds. The value of this field continually increases regardless of whether the LSA is transmitted over a link or saved in an LSDB.

Field	Length	Description
Options	8 bits	<p>Available options:</p> <ul style="list-style-type: none"> • E: AS-External-LSAs can be flooded. • MC: IP multicast packets are forwarded. • N/P: Type 7 LSAs are processed. • DC: Demand circuits are processed. • DN: It prevents loops on an MPLS VPN. When a PE sends Type 3, Type 5, or Type 7 LSAs to a CE, the DN bit needs to be set. When other PEs receive the LSAs from the CE, they cannot use these LSAs for OSPF route calculation. • O: It indicates whether the originating router supports Opaque LSAs (Type 9, Type 10, and Type 11).
LS type	8 bits	<p>LSA type:</p> <ul style="list-style-type: none"> • Type 1: Router-LSA • Type 2: Network-LSA • Type 3: Network-Summary-LSA • Type 4: ASBR-Summary-LSA • Type 5: AS-External-LSA • Type 7: NSSA-LSA
Link State ID	32 bits	This field, together with the LS type field, uniquely identifies an LSA in a routing domain.
Advertising Router	32 bits	Router ID of the router that generates the LSA.
LS sequence number	32 bits	Sequence number of the LSA. Other routers can determine which LSA is the latest according to this value.
LS checksum	16 bits	Checksum of all fields except the LS age field.
length	16 bits	Length of the LSA including the LSA header, in bytes.

Router-LSA

Router-LSA (Type 1): describes the link states and cost of a router. It is generated by each router and advertised in the area to which the router belongs. [Figure 1-107](#) shows the Router-LSA format.

Figure 1-107 Router-LSA format

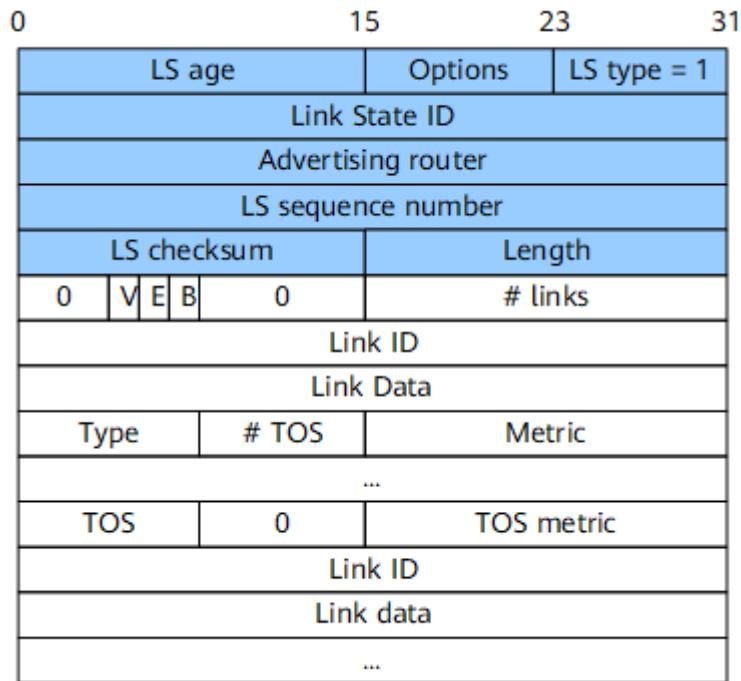


Table 1-41 Router-LSA fields

Field	Length	Description
Link State ID	32 bits	Router ID of the device that generates the LSA.
V (virtual link)	1 bit	If the router that generates the LSA is an endpoint of a virtual link, this field is set to 1. In other cases, this field is set to 0.
E (external)	1 bit	If the router that generates the LSA is an AS boundary router (ASBR), this field is set to 1. In other cases, this field is set to 0.
B (border)	1 bit	If the router that generates the LSA is an area border router (ABR), this field is set to 1. In other cases, this field is set to 0.
# links	16 bits	Number of links and interfaces described in the LSA, including all links and interfaces in the area to which the router belongs.
Link ID	32 bits	Object to which the router is connected. The value depends on the connection type. <ul style="list-style-type: none"> ● 1: router ID ● 2: interface IP address of the DR ● 3: network segment or subnet number ● 4: router ID of the neighbor on a virtual link

Field	Length	Description
Link Data	32 bits	Link data. The value varies according to the link type: <ul style="list-style-type: none"> • Unnumbered P2P: interface index • Stub network: subnet mask • Other connections: IP address of the router interface
Type	8 bits	Basic description of the router connection: <ul style="list-style-type: none"> • 1: connected to another router in P2P mode • 2: connected to a transit network • 3: connected to a stub network • 4: virtual link
# ToS	8 bits	Type of service (ToS) count.
metric	16 bits	Link cost.
ToS	8 bits	Type of service.
ToS metric	16 bits	Metric for the specified ToS.

Network-LSA

Network-LSA (Type 2): is generated by the DR on a broadcast or non-broadcast multiple access (NBMA) network. A Network-LSA records the router IDs of all routers on the network and describes the link states of all routers on the local network segment. It is advertised within the area to which the DR belongs. [Figure 1-108](#) shows the Network-LSA format.

Figure 1-108 Network-LSA format

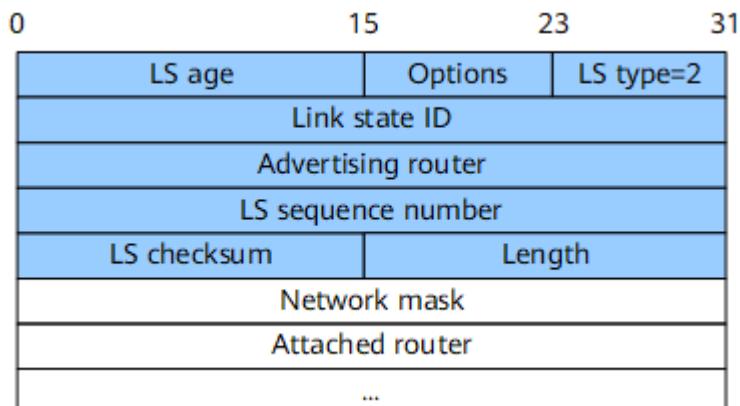


Table 1-42 Network-LSA fields

Field	Length	Description
Link State ID	32 bits	Interface IP address of the DR
Network Mask	32 bits	Mask of the broadcast or NBMA network
Attached Router	32 bits	Router IDs of all routers connected to the same network, including the router ID of the DR.

Summary-LSA

Network-Summary-LSA (Type 3): describes routes on a network segment in an area and is advertised to other related areas.

ASBR-Summary-LSA (Type 4): describes routes to an ASBR and is advertised to all related areas except the area to which the ASBR belongs.

Type 3 and Type 4 LSAs have the same format and are generated by ABRs. [Figure 1-109](#) shows the Summary-LSA format.

Figure 1-109 Summary-LSA format

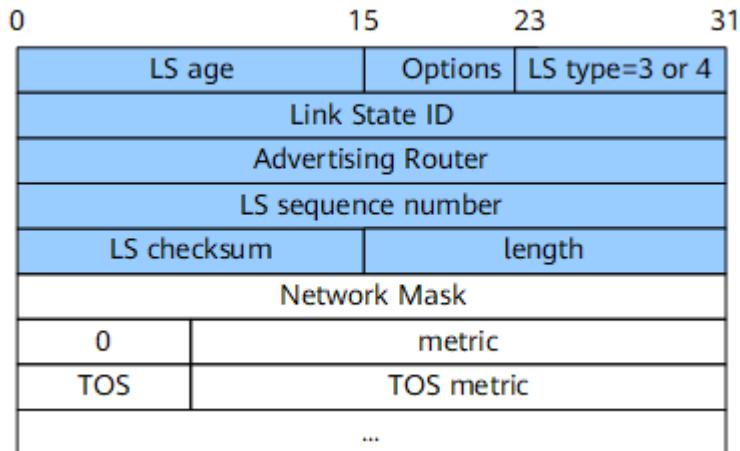


Table 1-43 Network-Summary-LSA fields

Field	Length	Description
Link State ID	32 bits	Advertised network address.
Network Mask	32 bits	Mask of the broadcast or NBMA network
metric	24 bits	Cost of the route to the destination address.
ToS	8 bits	Type of service.

Field	Length	Description
ToS metric	24 bits	Metric for the specified ToS.

 NOTE

When default routes are advertised, both the Link State ID and Network Mask fields are set to 0.0.0.0.

Table 1-44 ASBR-Summary-LSA fields

Field	Length	Description
Link State ID	32 bits	Router ID of the ASBR.
Network Mask	32 bits	This field is meaningless and is set to 0.0.0.0.
metric	24 bits	Cost of the route to the destination address.
ToS	8 bits	Type of service.
ToS metric	24 bits	Metric for the specified ToS.

AS-External-LSA

An AS-External-LSA (Type 5) describes AS external routes. AS-External-LSAs are generated by an ASBR. Among the five types of LSAs, only AS-External-LSAs can be advertised to all areas except stub areas and not-so-stubby areas (NSSAs).

[Figure 1-110](#) shows the AS-External-LSA format.

[Figure 1-110](#) AS-External-LSA format

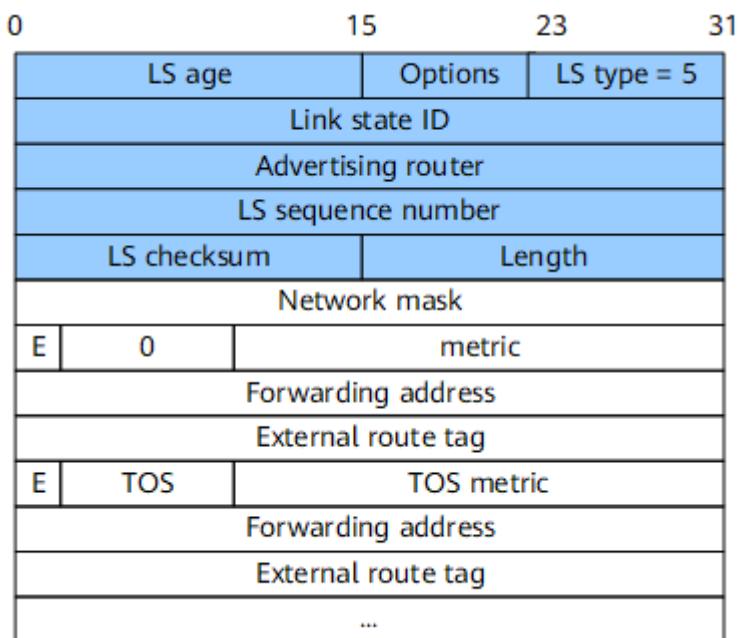


Table 1-45 AS-External-LSA fields

Field	Length	Description
Link State ID	32 bits	Advertised network address.
Network Mask	32 bits	Mask of the advertised destination address.
E	1 bit	Type of the external route. The values are as follows: <ul style="list-style-type: none">• 0: Type 1 external route• 1: Type 2 external route
metric	24 bits	Cost of the route to the destination address.
Forwarding Address	32 bits	Packets destined for the advertised destination address are forwarded to the address specified by this field.
External Route Tag	32 bits	Tag added to the external route. This field is not used by the OSPF protocol itself. It can be used to manage external routes.
ToS	8 bits	Type of service.
ToS metric	24 bits	Metric for the specified ToS.

 **NOTE**

If Type 5 LSAs are used to advertise default routes, both the Link State ID and Network Mask fields are set to 0.0.0.0.

Routing Loop Detection for Routes Imported to OSPF

Routes of an OSPF process can be imported to another OSPF process or the process of another protocol (such as IS-IS or BGP) for redistribution. However, if a device that performs such a route import is incorrectly configured, routing loops may occur. OSPF can use the routing loop detection function to detect routing loops.

Related Concepts

Redistribute ID

IS-IS uses a system ID as a redistribution identifier, OSPF and OSPFv3 use a router ID + process ID as a redistribution identifier, and BGP uses a VrfID + random number as a redistribution identifier. For ease of understanding, the redistribution identifiers of different protocols are all called Redistribute IDs. When routes are distributed, the information carried in the routes contains Redistribute IDs.

Redistribute List

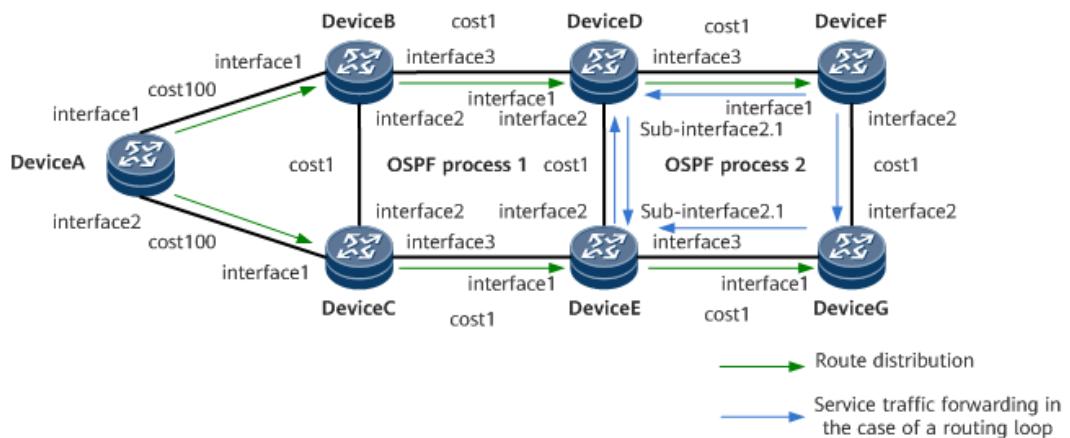
A Redistribute list may consist of multiple Redistribute IDs. Each Redistribute list of BGP contains a maximum of four Redistribute IDs, and each Redistribute list of

any other routing protocol contains a maximum of two Redistribute IDs. When the number of Redistribute IDs exceeds the corresponding limit, the old ones are discarded according to the sequence in which Redistribute IDs are added.

Cause (OSPF Inter-Process Mutual Route Import)

In [Figure 1-111](#), DeviceA, DeviceB, and DeviceC run OSPF process 1; DeviceF and DeviceG run OSPF process 2; DeviceD and DeviceE run both of the processes. Route import between OSPF process 1 and OSPF process 2 is configured on DeviceD and DeviceE. DeviceE imports routes from OSPF process 1 to OSPF process 2 for distribution. DeviceD imports the routes from OSPF process 2 to OSPF process 1 so that the routes are re-distributed back to OSPF process 1. As the costs of the routes newly distributed by DeviceD are smaller, they are preferentially selected by OSPF process 1, resulting in routing loops.

Figure 1-111 Typical network diagram of OSPF inter-process mutual route import

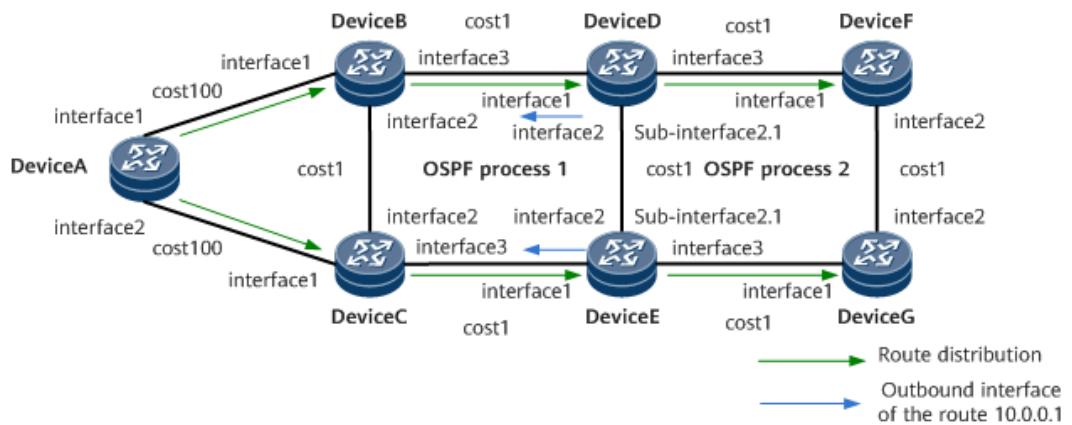


Take the route distributed by DeviceA as an example. A stable routing loop is formed through the following process:

Phase 1

On the network shown in [Figure 1-112](#), OSPF process 1 on DeviceA imports the static route 10.0.0.1 and floods a Type 5 AS-External-LSA in OSPF process 1. After receiving the LSA, OSPF process 1 on DeviceD and OSPF process 1 on DeviceE each calculate a route to 10.0.0.1, with the outbound interfaces being interface1 on DeviceD and interface1 on DeviceE, respectively, and the cost being 102. At this point, the routes to 10.0.0.1 in OSPF process 1 in the routing tables of DeviceD and DeviceE are active.

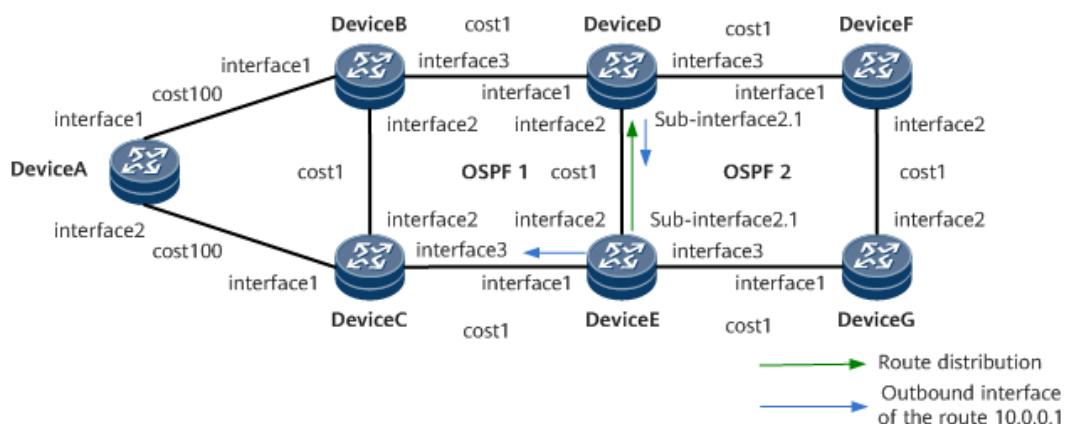
Figure 1-112 Phase 1



Phase 2

In [Figure 1-113](#), DeviceD is configured to import routes from OSPF process 1 to OSPF process 2, and DeviceE is configured to import routes from OSPF process 1 to OSPF process 2. No route-policy is configured for the import, or the configured route-policy is improper. For example, OSPF process 2 on DeviceE imports routes from OSPF process 1 and then floods a Type 5 AS-External-LSA in OSPF process 2. After receiving the LSA, OSPF process 2 on DeviceD calculates a route to 10.0.0.1, with the cost being 2, which is smaller than that (102) of the route calculated by OSPF process 1. As a result, the active route to 10.0.0.1 in the routing table of DeviceD is switched from the one calculated by OSPF process 1 to the one calculated by OSPF process 2, and the outbound interface of the route is sub-interface2.1.

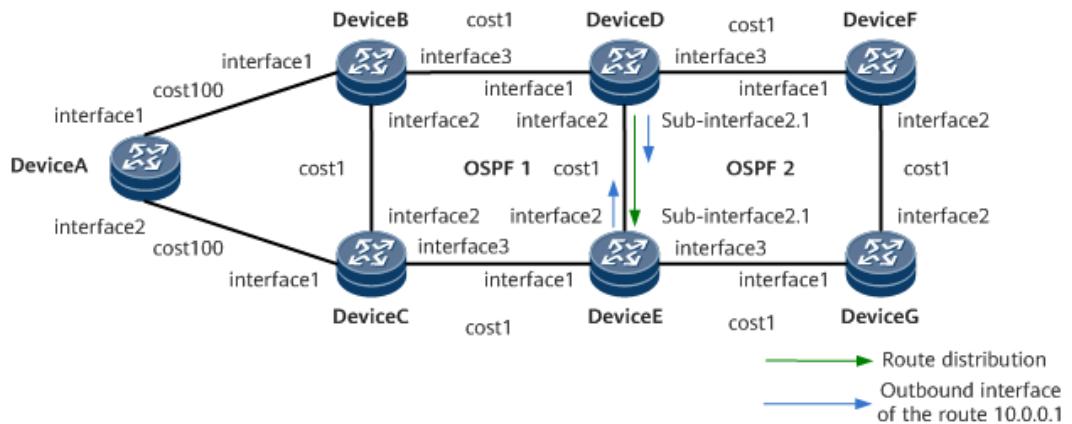
Figure 1-113 Phase 2



Phase 3

In [Figure 1-114](#), DeviceD imports the route from OSPF process 2 to OSPF process 1 and floods a Type 5 AS-External LSA in OSPF process 1. After receiving the LSA, OSPF process 1 on DeviceE recalculates the route to 10.0.0.1. The cost of the route becomes 2, which is smaller than that of the previously calculated route. Therefore, the route to 10.0.0.1 in OSPF process 1 on DeviceE is changed to the route distributed by DeviceD, and the outbound interface is interface 2.

Figure 1-114 Phase 3

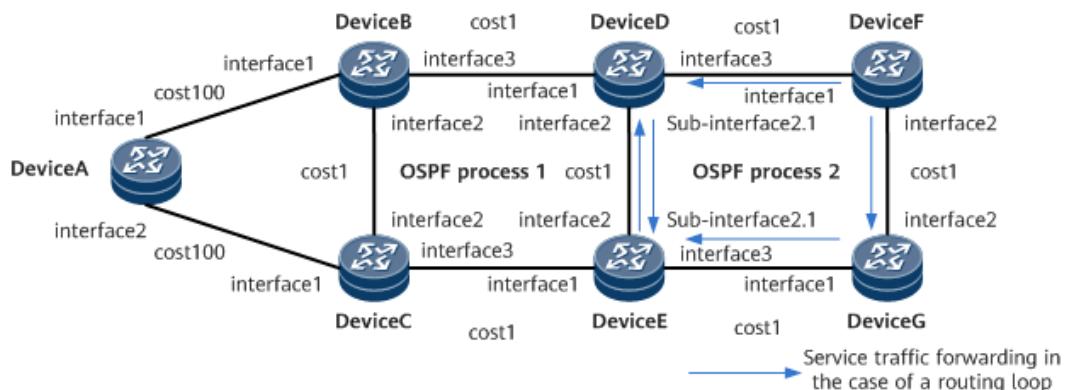


Phase 4

After the route to 10.0.0.1 on DeviceE is updated, OSPF process 2 still imports the route from OSPF process 1 as the route remains active, and continues to distribute/update a Type 5 AS-External-LSA.

As a result, a stable routing loop is formed. Assuming that traffic is injected from DeviceF, **Figure 1-115** shows the traffic flow when the routing loop occurs.

Figure 1-115 Traffic flow when a routing loop occurs



Implementation (OSPF Inter-Process Mutual Route Import)

Routing loop detection for the routes imported between OSPF processes can resolve the routing loops in the preceding scenario.

When distributing a Type 5 AS-External-LSA for an imported route, OSPF also uses a Type 11 extended prefix Opaque LSA to distribute to other devices the Redistribute ID of the device that redistributes the imported route. If the route is redistributed by different protocols through multiple devices, the Redistribute IDs of these protocols on the devices are distributed through a Type 11 extended prefix Opaque LSA. When receiving the Type 11 extended prefix Opaque LSA, a route calculation device saves the Redistribute ID and route information of the route redistribution device. When another process of a route calculation device imports the route, the device checks whether a routing loop occurs according to the route redistribution information. If a routing loop occurs, the device attaches a large route cost to the AS-External-LSA for the imported route so that other

devices preferentially select other paths after learning the route. This prevents routing loops.

Figure 1-116 Typical networking of route import to OSPF

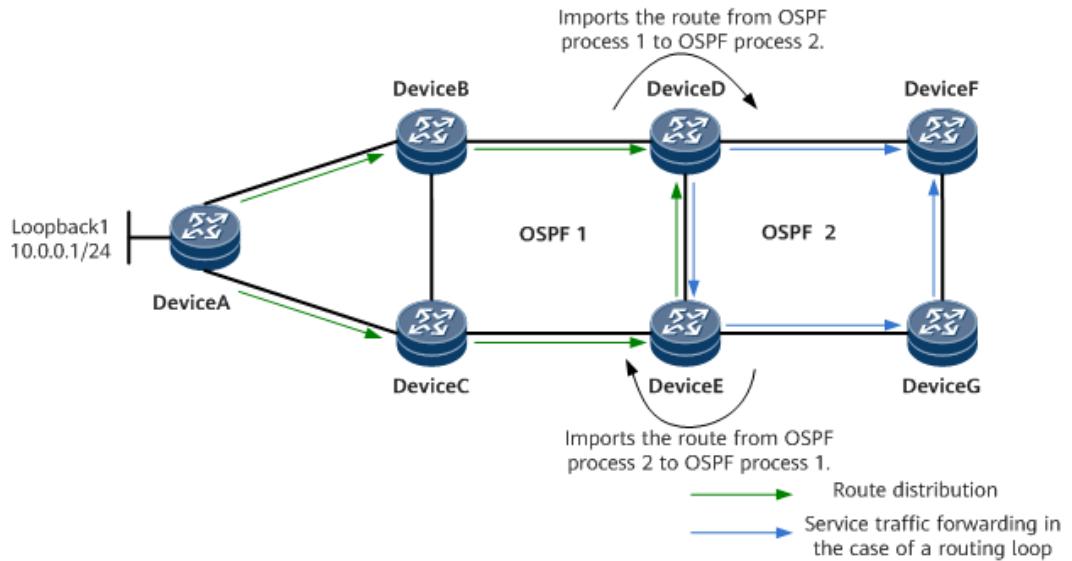


Figure 1-116 is used to describe how a routing loop is detected and resolved.

1. DeviceA distributes its locally originated route 10.0.0.1/24 to DeviceB.
2. DeviceD learns the routes advertised by DeviceB through OSPF process 1. When importing routes from OSPF process 1 to OSPF process 2, DeviceD adds Redistribute ID information to the routes to be distributed through Type 11 extended prefix Opaque LSAs. After learning the Redistribute ID information, DeviceE saves the information in its routing table.
3. DeviceE imports routes from OSPF process 2 to OSPF process 1 and redistributes the routes through OSPF process 1. The corresponding Type 11 extended prefix Opaque LSAs contain the Redistribute ID of OSPF process 1 on DeviceE and the Redistribute ID of OSPF process 2 on DeviceD.
4. OSPF process 1 on DeviceD learns the Redistribute list corresponding to the route distributed by DeviceE and saves the Redistribute list in the routing table. When importing the route from OSPF process 1 to OSPF process 2, DeviceD finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. OSPF process 2 on DeviceD distributes a large cost when redistributing the route so that other devices preferentially select other paths after learning the route. This prevents routing loops.

 NOTE

In the preceding typical networking:

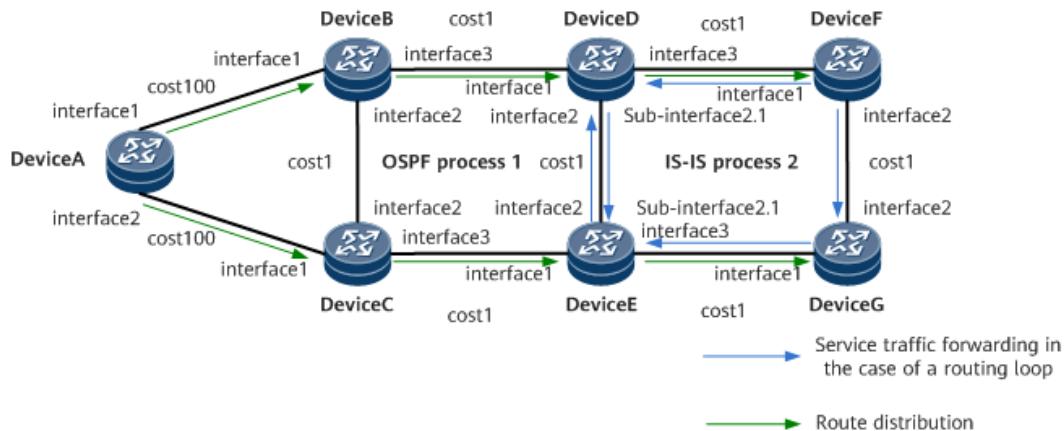
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

Cause (Mutual Route Import Between OSPF and IS-IS)

On the network shown in [Figure 1-117](#), DeviceA, DeviceB, and DeviceC run OSPF process 1, DeviceF and DeviceG run IS-IS process 2, and DeviceD and DeviceE run both processes. Route import between OSPF process 1 and IS-IS process 2 is configured on DeviceD and DeviceE. The ASE routes distributed by OSPF process 1 on DeviceE are re-distributed back to OSPF process 1 on DeviceD through IS-IS process 2. As the costs of the routes newly distributed by DeviceD are smaller, they are preferentially selected by OSPF process 1, resulting in routing loops.

Figure 1-117 Traffic flow when a routing loop occurs during route import between OSPF and IS-IS



Implementation (Mutual Route Import Between OSPF and IS-IS)

The following uses the networking shown in [Figure 1-117](#) as an example to describe how a routing loop is detected and resolved.

1. DeviceD learns the route distributed by DeviceB through OSPF process 1 and imports the route from OSPF process 1 to IS-IS process 2. When IS-IS process 2 on DeviceD distributes route information, it uses the extended prefix sub-TLV to distribute the Redistribute ID of IS-IS process 2 through an LSP. IS-IS process 2 on DeviceE learns the route distributed by DeviceD and saves the Redistribute ID distributed by IS-IS process 2 on DeviceD to the routing table during route calculation.
2. DeviceE imports the route from IS-IS process 2 to OSPF process 1 and uses an E-AS-External-LSA to distribute the Redistribute ID of OSPF process 1 on

DeviceE when distributing route information. Similarly, after OSPF process 1 on DeviceD learns the route from DeviceE, DeviceD saves the Redistribute ID distributed by OSPF process 1 on DeviceE to the routing table during route calculation.

3. When importing the route from OSPF process 1 to IS-IS process 2, DeviceD finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. IS-IS process 2 on DeviceD distributes a large cost when distributing the imported route. Because IS-IS has a higher preference than OSPF ASE, this does not affect the route selection result or resolve the routing loop.
4. DeviceE imports the route from IS-IS process 2 to OSPF process 1, finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. OSPF process 1 on DeviceE distributes a large cost when distributing the imported route so that other devices preferentially select other paths after learning the route. This prevents routing loops.

NOTE

In the preceding typical networking:

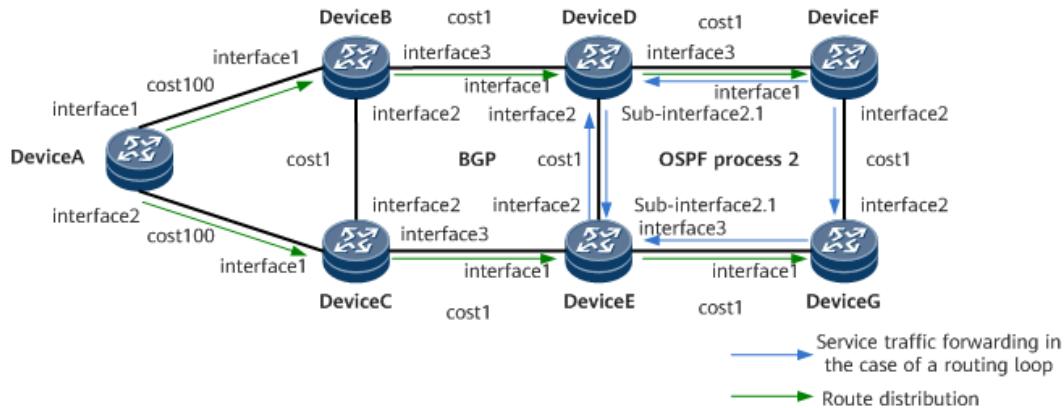
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

Cause (Mutual Route Import Between OSPF and BGP)

On the network shown in [Figure 1-118](#), DeviceA, DeviceB, and DeviceC run a BGP process, DeviceF and DeviceG run OSPF process 2, and DeviceD and DeviceE run both processes. A BGP peer relationship is established between DeviceD and DeviceE. Route import between BGP and OSPF process 2 is configured on DeviceD and DeviceE. DeviceE imports routes from BGP to OSPF process 2 for distribution. DeviceD imports routes from OSPF process 2 to BGP so that the routes are re-distributed back to BGP. Because no route-policy is configured for the import or the configured route-policy is improper, the route newly distributed by DeviceD may be selected as the optimal route by BGP, causing a routing loop.

Figure 1-118 Traffic flow when a routing loop occurs during route import between OSPF and BGP



Implementation (Mutual Route Import Between OSPF and BGP)

The following uses the networking shown in [Figure 1-118](#) as an example to describe how a routing loop is detected and resolved.

1. DeviceD learns the route distributed by DeviceB through BGP and imports the BGP route to OSPF process 2. When DeviceD distributes the imported route through OSPF process 2, it uses a Type 11 extended prefix Opaque LSA to distribute the Redistribute ID of OSPF process 2 on DeviceD. DeviceE learns the route distributed by DeviceD through OSPF process 2 and saves the Redistribute List distributed by DeviceD through OSPF process 2 to the routing table when calculating routes.
2. After DeviceE imports routes from OSPF process 2 to BGP, the Redistribute ID information of OSPF process 2 is inherited by the generated BGP routes, and routing loop attributes are generated and distributed. After BGP on DeviceD learns the routes distributed by DeviceE, DeviceD saves the routing loop attributes distributed by BGP on DeviceE in the routing table for route calculation.
3. When importing the routes from BGP to OSPF process 2, DeviceD finds that the Redistribute list of the routes contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. OSPF process 2 on DeviceD distributes a large link cost when distributing the imported route. Because OSPF has a higher preference than BGP, this does not affect the route selection result or resolve the routing loop.
4. After learning the route distributed by OSPF on DeviceD, DeviceE imports the route to BGP. Upon finding that the Redistribute list of the route contains its own Redistribute ID, DeviceE considers that a routing loop is detected and reports an alarm. When BGP on DeviceE distributes the route, it reduces the preference of the route. In this way, other devices preferentially select other paths after learning this route, preventing routing loops.

NOTE

In the preceding typical networking:

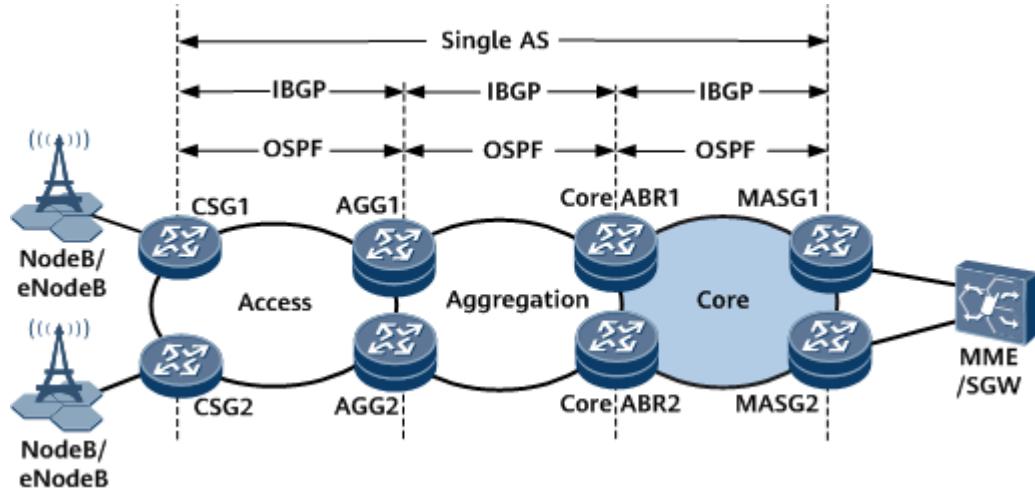
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

Usage Scenario

Figure 1-119 shows a typical intra-AS seamless MPLS network. If the OSPF process deployed at the access layer differs from that deployed at the aggregation layer, OSPF inter-process mutual route import is usually configured on AGGs so that routes can be leaked between the access and aggregation layers. In this case, a routing loop may occur between AGG1 and AGG2. If routing loop detection for OSPF is configured on AGG1 and AGG2, routing loops can be quickly detected and other routes can be preferentially selected, preventing routing loops.

Figure 1-119 Routing protocol deployment on the intra-AS seamless MPLS network



1.1.4.2 OSPF Configuration

By building OSPF networks, you can enable OSPF to discover and calculate routes in an AS. OSPF is applicable to large-scale networks with hundreds of routers.

1.1.4.2.1 Overview of OSPF

Open Shortest Path First (OSPF), which is developed by the IETF, is a link-state IGP. OSPF is widely used in access networks and MANs.

Definition

Open Shortest Path First (OSPF) is a link-state Interior Gateway Protocol (IGP) developed by the Internet Engineering Task Force (IETF).

OSPF version 2 (OSPFv2) is intended for IPv4. OSPF version 3 (OSPFv3) is intended for IPv6.

NOTE

In this document, OSPF refers to OSPFv2, unless otherwise stated.

Purpose

Before the emergence of OSPF, the Routing Information Protocol (RIP) was widely used as an IGP on networks. RIP is a distance-vector routing protocol. Due to its slow convergence, routing loops, and poor scalability, RIP is gradually being replaced with OSPF.

Typical IGPs include RIP, OSPF, and Intermediate System to Intermediate System (IS-IS). **Table 1-46** describes differences among the three typical IGPs.

Table 1-46 Differences among RIP, OSPF, and IS-IS

Item	RIP	OSPF	IS-IS
Protocol type	IP layer protocol	IP layer protocol	Link layer protocol
Application scope	Applies to small networks with simple architectures, such as campus networks.	Applies to medium-sized networks with several hundred routers supported, such as enterprise networks.	Applies to large networks, such as Internet service provider (ISP) networks.
Routing algorithm	Uses a distance-vector algorithm and exchanges routing information over the User Datagram Protocol (UDP).	Uses the shortest path first (SPF) algorithm to generate a shortest path tree (SPT) based on the network topology, calculates shortest paths to all destinations, and exchanges routing information over IP.	Uses the SPF algorithm to generate an SPT based on the network topology, calculates shortest paths to all destinations, and exchanges routing information over IP. The SPF algorithm runs separately in Level-1 and Level-2 databases.
Route convergence speed	Slow	Less than 1 second	Less than 1 second

Item	RIP	OSPF	IS-IS
Scalability	Not supported	Supported by partitioning a network into areas	Supported by defining router levels

Benefits

OSPF offers the following benefits:

- Wide application scope: OSPF applies to medium-sized networks with several hundred routers, such as enterprise networks.
- Network masks: OSPF packets can carry masks, and therefore the packet length is not limited by natural IP masks. OSPF can process variable length subnet masks (VLSMs).
- Fast convergence: When the network topology changes, OSPF immediately sends link state update (LSU) packets to synchronize the changes to the link state databases (LSDBs) of all routers in the same autonomous system (AS).
- Loop-free routing: OSPF uses the SPF algorithm to calculate loop-free routes based on the collected link status.
- Area partitioning: OSPF allows an AS to be partitioned into areas, which simplifies management. Routing information transmitted between areas is summarized, which reduces network bandwidth consumption.
- Equal-cost routes: OSPF supports multiple equal-cost routes to the same destination.
- Hierarchical routing: OSPF uses intra-area routes, inter-area routes, Type 1 external routes, and Type 2 external routes, which are listed in descending order of priority.
- Authentication: OSPF supports area-based and interface-based packet authentication, which ensures packet exchange security.
- Multicast: OSPF uses multicast addresses to send packets on certain types of links, which minimizes the impact on other devices.

1.1.4.2.2 Configuration Precautions for OSPF

Feature Requirements

Table 1-47 Feature requirements

Feature Requirements	Series	Models
<p>1. When the primary next hop is a TE tunnel, microloop avoidance is not calculated.</p> <p>2. For routes with a single route source, load balancing is not supported during microloop avoidance. During microloop avoidance, only one next hop with a strict label stack or a single outbound interface is delivered to the product. After the microloop avoidance period expires, normal convergence entries are delivered.</p> <p>If multiple nodes advertise the same route, load balancing may be performed between the next hop with microloop avoidance and the common next hop.</p> <p>3. The length of the label stack is limited based on the interface capability of the board. The maximum length is 10. If the number of hops exceeds 10, the system does not enter the microloop avoidance state and directly delivers the converged path.</p> <p>4. No microloop avoidance entry is generated on the interface whose label stack capability is 0 or 255.</p> <p>5. Private networks are not supported.</p> <p>6. IP route microloop avoidance is not supported.</p> <p>7. In a broadcast network switchback scenario, if there is more than one neighbor on the switchback link, the device cannot wait for the neighbor labels of multiple neighbors. The microloop avoidance may fail.</p> <p>8. SRs on the entire network must be able to advertise neighbor labels,</p> <p>9. If the link for traffic switchback lacks a neighbor label, path calculation is delayed after traffic switchback. If no neighbor label is received within a certain period of time, microloop avoidance fails. During the delay calculation, a microloop avoidance label stack is delivered for the current path.</p> <p>10. Multiple nodes advertise the same route prefix. When the optimal source node changes</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>due to a switchover or switchback, the next hop of the prefix changes. If the SR next hop of the new optimal source node remains unchanged, the microloop avoidance path is not calculated.</p> <p>11. Parallel tags are not supported.</p>		

Feature Requirements	Series	Models
<p>1. Rules for selecting a node label advertised by an OSPF TI-LFA P node are as follows:</p> <ul style="list-style-type: none"> a) The prefix label with the N flag advertised by the P node is preferred. For inter-area or imported routes, the prefix label with the A flag must be set. b) Prefix label of the smallest SID of the prefix with a single route source on the P node c) The P node preferentially selects the route prefix with a single route source. If the route prefix has multiple route sources, ensure that the P node is the only optimal source. If routes with a single route source and prefix labels for load balancing cannot be selected, the prefix labels for load balancing advertised by multiple nodes can be preferentially selected as P SIDs. However, the node SIDs of Ps that advertise the same route and work in load balancing mode can only be used to protect anycast prefixes, and the PQs of anycast prefixes must be the same. d) A node that does not support SR cannot function as a P node. A node that does not advertise prefix labels cannot function as a P node. <p>2. In the case of IP load balancing, OSPF does not calculate the TI-FRR backup path even if there is only one primary prefix label tunnel.</p> <p>3. OSPF anycast prefix label, which is used to calculate the TI-FRR backup path.</p> <ul style="list-style-type: none"> a) Anycast prefix labels and IP routes share virtual nodes when multiple nodes advertise the same routes. b) A maximum of 256 virtual nodes are supported. c) The number of anycast prefix label sources cannot exceed 16. <p>4. In TI-LFA scenarios, TI-LFA calculates routes in RLFA scenarios (that is, PQs overlap and PQ nodes are not neighboring nodes). If no node label is configured for PQ nodes, TI-LFA calculation fails. This is a configuration error and remote LFA calculation is not performed.</p> <p>5. If the cost of the backup path is greater than or equal to 65535, OSPF considers the path unreachable and does not select the path</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
whose cost is greater than or equal to 65535 as the backup path.		
After SR LSP strict check is configured, the SR capability of a node is strictly checked when LSPs are calculated from the current node to all nodes. If SR-LDP interworking has been deployed on the network, OSPF SR-LDP interworking may fail, causing traffic interruptions.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
OSPF checks whether a neighbor in the full state exists in the backbone area before advertising a default route to the stub area.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
When a device in an NSSA generates an NSSA LSA based on an imported external route, the device preferentially uses the IP address of a loopback interface in the NSSA as the forwarding address (FA). If no loopback interfaces exist in the NSSA, the device selects the IP address of a non-loopback interface. As a result, the downstream device may fail to implement load balancing using routes even when links with the same cost exist.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
OSPF advertises default routes to an NSSA only when neighbor relationships in the Full state exist in the backbone area or default routes of another protocol or of another OSPF process exist in the same VPN instance on the device, and the nssa default-route-advertise command is run.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>1. The mapping SID priority is not supported. All mapping SIDs are processed based on the default priority.</p> <p>2. On the TI-LFA protection path, all devices from the source node to the Q node must support segment routing, and an LDP path must be available from the Q node to the destination node.</p> <p>3. Anycast FRR cannot be implemented for SR routes mapped to mapping SIDs. Anycast protection cannot be implemented between SR routes mapped to mapping SIDs or between SR routes mapped to mapping SIDs and common SR routes.</p> <p>4. SR routes mapped to mapping SIDs do not support remote anti-microloop.</p> <p>5. In cross-process and cross-protocol scenarios, the ASBR functions as an interworking stitching node, and the MappingSid mapping needs to be configured on the process to which the route is imported. Properly plan configurations.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>Penultimate hop popping (PHP) causes SR interworking to fail.</p> <p>Plan the configuration properly.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>SR interworking takes effect only when a mapping SID with an IP prefix with a 32-bit mask is used. SR interworking does not take effect for mapping SIDs with IP prefixes of non-32-bit masks. SR interworking does not take effect for mapping SIDs with IP prefixes of non-32-bit masks.</p> <p>Plan the configuration properly.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>In SR-LDP interworking scenarios, FRR does not take effect in some scenarios. For example:</p> <p>In the SR-to-LDP direction, the Q node and all NEs on the path before the Q node must have the SR capability,</p> <p>In the LDP-to-SR interworking FRR scenario, PQ node and all NEs on the path before the PQ path must have the LDP capability.</p> <p>Plan the configuration properly.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>In an OSPF loop detection scenario, when the maximum cost is advertised due to a loop, the apply cost command does not take effect.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>The network types of the interfaces on both ends of an OSPF neighbor relationship must be the same so that the interfaces can learn routes after the neighbor relationship is established. Otherwise, the neighbor relationship cannot be established and routes cannot be calculated.</p> <p>You are advised to use the same network type at both ends.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>In OSPF session CAR scenarios, OSPF neighbors in multiple areas on the same interface cannot be distinguished on the forwarding plane. Therefore, they share the same session CAR entry.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>Constraints on the association between BFD and cost values of OSPF broadcast links</p> <p>1) When there is only one neighbor, the neighbor is not deleted when BFD goes Down, and the OSPF LSDB component is notified that the neighbor enters the BFD-associated cost state.</p> <p>2) This function is not supported when there are multiple neighbors.</p> <p>When the number of neighbors changes from one to multiple, the neighbor in the BFD Down state is deleted and the OSPF LSDB component is notified that the neighbor is deleted.</p> <p>It is recommended that only one neighbor relationship be established on an OSPF broadcast network.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>SBFD does not take effect in SR-LDP interworking scenarios because SBFD does not take effect in LDP scenarios.</p> <p>Plan the configuration properly.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>In an OSPF loop detection scenario, after a loop is removed, the loop cannot be automatically removed and the alarm cannot be automatically cleared. Manual intervention is required. After the routing policy or route tag is correctly configured, run the clear route loop-detect ospf alarm-state command to exit the loop state and clear the alarm.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>In OSPF loop detection scenarios, only loops caused by inter-process route import between two devices can be detected. If more than two devices import routes between processes, loops cannot be detected.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>In OSPF loop detection scenarios, loop detection cannot be performed on aggregated routes or NSSA routes.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
In OSPF loop detection scenarios, incorrect detection may be triggered when a router ID conflict occurs, including intra-AS router ID conflict and inter-AS router ID conflict.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
In OSPF loop detection scenarios, a single process can perform loop detection for a maximum of 100,000 routes. If the number of routes exceeds 100,000, loop detection cannot be performed.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
OSPF loop detection depends on the Opaque capability of OSPF in a process. If the Opaque capability is not enabled, OSPF loop detection does not take effect. In this case, you need to manually enable the Opaque capability of OSPF.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
In OSPF loop detection scenarios, loops can be detected for the default route advertised by the default-route-advertise command but self-healing cannot be achieved.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
After the device is restarted, if the BFD session of the local device or its neighbor is in Admin Down state, the OSPF status is not affected. When the BFD session is renegotiated, if the BFD detection status reported by BFD is Down (used to be Up), the OSPF neighbor is set to Down. In other cases, the OSPF status is not affected.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

1.1.4.2.3 Configuring Basic OSPF Functions

Before constructing OSPF networks, you need to configure basic OSPF functions.

Usage Scenario

When OSPF is configured on multiple routers in the same area, most configuration data, such as the timer, filter, and aggregation, must be planned uniformly in the area. Incorrect configurations may cause neighboring routers to

fail to send messages to each other or even causing routing information congestion and self-loops.

The OSPF-relevant commands that are configured in the interface view take effect regardless of whether OSPF is enabled. After OSPF is disabled, the OSPF-relevant commands also exist on interfaces.

Pre-configuration Tasks

Before configuring basic OSPF functions, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.

(Optional) Configuring OSPF Short-Timeout Neighbor Enhancement

OSPF short-timeout neighbor enhancement improves neighbor stability in scenarios where there are many short-timeout neighbors.

Usage Scenario

If the configured interval at which an interface sends Hello packets is less than 10s, many OSPF short-timeout neighbors may exist. (For details about setting the interval, see [Setting the Interval at Which Hello Packets Are Sent](#).) In this case, if the device has poor performance, and the CPU usage is high or a switchover is performed, the neighbors may be unstable due to scheduling issues. To improve the stability of short-timeout neighbors in scenarios where there are more than 150 OSPF and OSPFv3 short-timeout neighbors, you can configure the short-timeout neighbor enhancement function, which increases the OSPF short-timeout neighbor specification.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf short-timeout neighbor enhancement**

OSPF short-timeout neighbor enhancement is configured.



- NOTE**
- If the device has more than 150 short-timeout neighbors and this function is configured or deleted when the CPU usage is high, the neighbors may be intermittently disconnected. If this function is required, you are advised to perform this step before [configuring an OSPF process](#).
 - After this step is performed, short-timeout neighbor enhancement takes effect for both OSPF and OSPFv3.

Step 3 Run **commit**

The configuration is committed.

----End

Enabling OSPF

After an OSPF process is created, a router ID is configured for the router, an interface on which OSPF runs and the area to which the interface belongs are specified, routes can be discovered and calculated in the AS.

Context

OSPF on the router requires a router ID. The router ID is a 32-bit unsigned integer, which uniquely identifies the router in the AS. To ensure the stability of OSPF, manually set the router ID of each router during network planning.

OSPF prevents the link state database (LSDB) size from unexpectedly growing by partitioning an AS into different areas. An area is regarded as a logical group, and each group is identified by an area ID. At the border of an area resides the router instead of a link. A network segment (or a link) belongs to only one area. The area to which each OSPF interface belongs must be specified.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id] [router-id router-id] ***

An OSPF process is started, and the OSPF view is displayed.

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports OSPF multi-instance. To run OSPF in a VPN instance, run the **ospf [process-id] [router-id router-id | vpn-instance vpn-instance-name] *** command.

- The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports OSPF multi-process. Processes can be classified by service type. The routers exchange packets regardless of process IDs.
- **router-id router-id** specifies the router ID of the router.

By default, the router automatically selects the IP address of an interface as the router ID. When configuring the router ID, ensure that the router ID is unique in an AS. You can configure the IP address of an interface as the router ID.

NOTE

Each router ID in an OSPF process must be unique. Otherwise, no OSPF neighbor relationships can be established, and routing information is incorrect.

If a router ID conflict occurs, perform either of the following operations:

- Run the **ospf router-id router-id** command to configure a new router ID.
- Run the **undo ospf router-id auto-recover disable** command to enable the router ID automatic recovery function. After the function is enabled, the system automatically allocates a new router ID if a router ID conflict occurs.

 NOTE

- If the automatic recovery function is enabled and a router ID conflict occurs between indirectly connected routers in one OSPF area, the conflicting router ID is replaced with a newly calculated one, regardless of whether the conflicting router ID was manually configured or automatically generated.
- The system can replace a router ID in a maximum of three attempts in case the router ID conflict persists.

After a router ID is replaced, the **reset ospf [process-id] process** command needs to be run to validate the new router ID.

- If a VPN instance is specified, the OSPF process belongs to this VPN instance. If a VPN instance is not specified, the OSPF process belongs to the public-network instance.

You can run the **description** command to configure a description for the OSPF process for easier identification.

Step 3 Run **area area-id**

The OSPF area view is displayed.

OSPF areas are classified as a backbone area (with area ID 0) or non-backbone areas. The backbone area forwards inter-area routing information. The routing information exchanged between non-backbone areas must be forwarded through the backbone area.

You can run the **description** command to configure a description for the OSPF area for easier identification.

Step 4 To configure OSPF, you can configure the network segments included in an area or enable OSPF on an interface.

- To configure the network segments included in an area, run the **network address wildcard-mask [description text]** command, in which **description** specifies the description of the area.

OSPF runs on an interface only when both of the following conditions are met:

- The mask length of the interface's IP address is greater than or equal to that specified in the **network** command.

 NOTE

If *wildcard-mask* in the **network** command are all 0s and the IP address of the interface is the same as the IP address specified in the **network address** command, OSPF is also enabled on the interface.

- The interface's primary IP address belongs to the network segment specified in the **network** command.

By default, OSPF uses a host route with a 32-bit mask to advertise the IP address of a loopback interface, regardless of the mask length configured for the IP address. To allow a loopback interface to advertise network-segment routes, its network type must be set to NBMA or broadcast in the interface view. For details on how to set the network type, see [Configuring Network Types for OSPF Interfaces](#).

- Enable OSPF on an interface.

- a. Run the **quit** command twice to return to the system view.
- b. Run the **interface interface-type interface-number** command to enter the interface view.
- c. Run the **ospf enable [process-id] area area-id** command to enable OSPF on the interface. The specified area ID can be a decimal integer or in the format of an IPv4 address. Regardless of the specified format, the area ID is displayed as an IPv4 address.

Step 5 Run commit

The configuration is committed.

----End

(Optional) Configuring an Interface to Fill in DD Packets with Its Own MTU

You can configure an interface to fill in the Interface MTU field of a DD packet with the interface MTU.

Context

To improve compatibility with a non-Huawei device, an OSPF-enabled Huawei device adds the MTU 0 in DD packets to be sent and does not check the MTUs in received DD packets, thereby allowing an OSPF neighbor relationship to be set up even if the two ends have different MTU settings.

However, under the default configuration, the non-Huawei device may discard an OSPF packet received from the Huawei device if the actual MTU in the packet is greater than the MTU of the non-Huawei device. If an LSU is discarded, an OSPF neighbor relationship can still be set up, but the routing information carried in the LSU fails to be learned, causing service interruption.

To resolve this issue, run the **ospf mtu-enable** command to configure an interface to add the actual MTU in DD packets to be sent and check whether the MTU in a received DD packet is greater than the local MTU. If the interface MTU settings of the local and remote ends are different, an OSPF neighbor relationship cannot enter the Full state. In this manner, MTU inconsistency can then be identified in time.

NOTICE

Setting the MTU in a DD packet will have the neighbor relationship reestablished.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run interface interface-type interface-number

The interface view is displayed.

Step 3 Run ospf mtu-enable

The interface is configured to fill in a DD packet with the interface MTU and check whether the MTU in the DD packet from the neighboring router exceeds the MTU of the local router.

Step 4 Run commit

The configuration is committed.

----End

(Optional) Creating OSPF Virtual Links

This section describes how to create logical links between backbone areas and to ensure the OSPF network connectivity.

Context

After OSPF areas are deployed, OSPF route updates between non-backbone areas are transmitted through the backbone area. Therefore, OSPF requires that all non-backbone areas be connected to a backbone area and backbone areas be connected as well. However, these requirements may not be met due to various limitations. OSPF virtual links can be configured to solve the problem.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ospf [*process-id*]

The OSPF view is displayed.

Step 3 Run area *area-id*

The OSPF area view is displayed.

Step 4 Run vlink-peer *router-id* [**dead** *dead-interval* | **hello** *hello-interval* | **retransmit** *retransmit-interval* | **trans-delay** *trans-delay-interval* | **smart-discover** | [**simple** [**plain** *plain-text* | [**cipher**] *cipher-text*] | { **md5** | **hmac-md5** | **hmac-sha256** } [**key-id** { **plain** *plain-text* | [**cipher**] *cipher-text* }] | **authentication-null** | **keychain** *keychain-name*]] *

A virtual link is created.

This command must be run on the other end of the virtual link as well.

The default parameter values are recommended when a virtual link is configured. If the network has special requirements, change the parameter values as required. Suggested parameter configurations are as follows:

- The smaller the **hello** value, the faster the router detects network topology changes, and the more network resources are consumed.
- If the value of **retransmit** is too small, unnecessary LSA retransmission may occur. It is recommended that you set a large value on a low-speed network.
- The authentication modes of a virtual link and the backbone area must be the same.

Step 5 Run commit

The configuration is committed.

----End

(Optional) Configuring the router to Comply with Route Selection Rules Defined in a Standard Protocol

You can configure the router to comply with the route selection rule defined in RFC 1583 or RFC 2328.

Context

RFC 2328 and RFC 1583 define route selection rules differently. After enabling OSPF on the router, you can configure the device to comply with route selection rules defined in either standard protocol as required. By default, the router complies with the route selection rules defined in RFC 1583. If you want the device to comply with the other protocol, you need to configure the device to comply with the rules defined in RFC 2328. Such configurations ensure that all OSPF-enabled devices in an AS comply with the same route selection rules defined in the same standard protocol.

If both intra-area and inter-area paths to an ASBR exist on a network, the default rules for selecting a path to the ASBR are as follows:

1. In RFC 1583 compatibility mode:
 - If the area IDs of the intra-area and inter-area paths to the ASBR are the same, intra-area paths are preferred.
 - If the area IDs of intra-area and inter-area paths to the ASBR are different, the path with the smallest cost is preferred; if their costs are the same, the path with the largest area ID is preferred.
2. In RFC 1583 non-compatibility mode:
 - If the area IDs of the intra-area and inter-area paths to the ASBR are the same and the paths belong to non-backbone areas, intra-area paths are preferred.
 - If the area IDs of the intra-area and inter-area paths to the ASBR are the same and the paths belong to the backbone area, the path with the smallest cost is preferred; if their costs are the same, load balancing is supported.
 - If the area IDs of the intra-area and inter-area paths to the ASBR are different, intra-area paths that belong to non-backbone areas are preferred; if intra-area paths belong to the backbone area, the path with the smallest cost is preferred; if their costs are the same, the path with the largest area ID is preferred.

 NOTE

If devices of different vendors or different series of devices of the same vendor are deployed on the same network, the rules for selecting a path to an ASBR among intra-area and inter-area paths may vary according to the mode (RFC 1583 compatibility mode or RFC 1583 non-compatibility mode). In this case, routing loops may occur. To prevent the routing loops, you can set the path selection rules to the default ones.

To prevent routing loops, ensure that all devices on the network use the same path selection rules. If adjustment is performed only on some devices, the adjustment fails to meet expectations. Therefore, exercise caution when adjusting path selection rules.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 (Optional) Run **rfc1583 compatible different-area-path prefer lower-cost**

The device is configured to comply with the default rules in RFC 1583 compatibility mode for selecting a path to an ASBR. That is, if the area IDs of intra-area and inter-area paths to the ASBR are different, the path with the smallest cost is preferred; if their costs are the same, the path with the largest area ID is preferred.

Step 4 Run **undo rfc1583 compatible**

The device is configured to comply with the route selection rule defined in RFC 2328, rather than RFC 1583.

Step 5 (Optional) Run **rfc1583 non-compatible backbone-area-path prefer intra**

The device is configured to comply with the default rules in RFC 1583 non-compatibility mode for selecting a path to an ASBR. That is, if the area IDs of the intra-area and inter-area paths to the ASBR are the same and the paths belong to the backbone area, intra-area paths are preferred.

Step 6 Run **commit**

The configuration is committed.

----End

(Optional) Setting the OSPF Preference

When multiple routing protocols discover routes to the same destination address, you can set the OSPF preference to control the route selection result.

Context

If a router runs multiple dynamic routing protocols at the same time, routing information may be shared and selected among the routing protocols. The system sets a priority for each routing protocol. When multiple routing protocols are used to select routes, the route selected by the routing protocol with a higher priority takes effect.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **preference [ase | inter | intra] { preference | { route-policy route-policy-name | route-filter route-filter-name } }***

A preference is set for OSPF.

- **ase**: indicates the AS external routes for which a preference is set.
- **inter**: indicates the inter-area routes for which a preference is set.
- **intra**: indicates the intra-area routes for which a preference is set.
- **preference**: specifies a preference value for OSPF routes. The smaller the value, the higher the priority.
- **route-policy-name**: specifies a route-policy to filter routes so that the preset preference is applied to the matched routes.
- **route-filter-name**: specifies a route-filter to filter routes so that the preset preference is applied to the matched routes.

Step 4 Run **commit**

The configuration is committed.

----End

(Optional) Restricting the Flooding of OSPF Update LSAs

To maintain stable OSPF neighbor relationships, you can restrict the flooding of update LSAs on a local device, so that its neighbors will not discard Hello packets due to a great number of update LSAs.

Context

If a local device has a large number of neighbors or needs to flood a large number of update LSAs, neighbor routers will receive a large number of update packets within a short time. In this case, neighbor routers are busy with processing the burst update packets and discard the Hello packets that are used to maintain neighbor relationships. As a result, the established OSPF neighbor relationships will be terminated. Then, the neighbor routers will exchange more packets to re-establish the neighbor relationships, exacerbating the problem of excessive packets.

To prevent this problem and ensure the stability of neighbor relationships, you can configure OSPF to restrict the flooding of update LSAs.

NOTE

The parameter settings of this command directly affect the flooding speed. If the parameters are improperly set, LSAs may not be synchronized in time, which affects the routing of the entire network. Therefore, this function is not recommended unless otherwise required.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **flooding-control [number number-value | timer-interval timer-interval-value]^{*}**

OSPF is configured to restrict the flooding of update LSAs.

Step 4 Run **commit**

The configuration is committed.

----End

(Optional) Configuring an Alarm Threshold for the Number of LSAs Learned by OSPF

You can configure an alarm threshold for the number of LSAs learned by OSPF so that an alarm is reported when this threshold is reached or exceeded.

Context

If the number of external routes that OSPF imports and advertises exceeds the routing table capacity of a device, the device may restart unexpectedly. To prevent this problem and ensure that the device runs properly, you can set an alarm threshold for the number of LSAs learned by OSPF and enable overload control.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **maximum received-lsa threshold value [overload-limit]**

An alarm threshold is configured for the number of LSAs learned by OSPF. You can determine whether to enable overload control as required.

- If an alarm threshold is configured but overload control is not, only an alarm is reported when this threshold is reached or exceeded.
- If both an alarm threshold and overload control are configured, an alarm is reported when this threshold is reached or exceeded. In addition, OSPF no longer learns new LSAs when the following conditions are met:
 - The number of LSAs learned by OSPF reaches or exceeds the alarm threshold.
 - The memory is in the danger state, and the memory used by the OSPF LSDB component ranks in the top 3.

- The **ospf memory-overload control** command configuration exists (the corresponding function is enabled by default).

Step 4 Run **commit**

The configuration is committed.

----End

(Optional) Configuring the Maximum Number of Packet Retransmission Attempts

When no response to DD packets, LSU packets, or LSR packets is received, the retransmission mechanism is used and the maximum number of packet retransmission attempts is set to prevent dead loops caused by repeated transmissions.

Context

If no response is received when the maximum number of packet retransmission attempts is reached, the neighbor relationship will be interrupted.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **retransmission-limit [max-number]** The maximum number of OSPF packet retransmission attempts is configured.

Step 4 Run **commit**

The configuration is committed.

----End

(Optional) Setting the Interval at Which LSAs Are Retransmitted Between Adjacent routers

You can set an appropriate interval at which LSAs are retransmitted based on network conditions in order to accelerate convergence.

Context

After sending an LSA to its neighbor, a device waits for a response. If the device does not receive an acknowledgement packet within the Nth LSA retransmission interval, the device retransmits the LSA to the neighbor.

First LSA retransmission interval = Configured interval at which LSAs are retransmitted, which is *interval*

Second LSA retransmission interval = Configured interval at which LSAs are retransmitted, which is *interval*

Third LSA retransmission interval = Configured interval at which LSAs are retransmitted, which is *interval*

Fourth LSA retransmission interval = Configured interval at which LSAs are retransmitted (*interval*) x 2

Fifth LSA retransmission interval = Configured interval at which LSAs are retransmitted (*interval*) x 2²

... ...

Nth LSA Retransmission Interval = Configured interval at which LSAs are retransmitted (*interval*) x 2⁽ⁿ⁻³⁾.

If *interval* x 2⁽ⁿ⁻³⁾ is greater than 30, the Nth LSA retransmission interval is 30.

If the configured interval at which LSAs are retransmitted (*interval*) is greater than 30, the Nth LSA retransmission interval equals *interval*.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

Step 3 Run **ospf timer retransmit interval**

An interval for retransmitting LSAs to the adjacent router is configured.

Set the LSA retransmission interval to a proper value. An excessively short interval will cause unnecessary retransmission. Generally, the interval should be longer than the time taken for round-trip packet transmission between two ends.

The default value is recommended.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring basic OSPF functions, verify information about OSPF neighbors, interfaces, and routes.

Prerequisites

Basic OSPF functions have been configured.

Procedure

- Run the **display ospf [process-id] abr-asbr [router-id]** command in any view to check information about the ABRs and ASBRs of OSPF.
- Run the **display ospf [process-id] cumulative** command in any view to check information about OSPF statistics.

- Run the **display ospf [process-id] peer** command in any view to check information about OSPF neighbors.
- Run the **display ospf [process-id] nexthop** command in any view to check information OSPF next hop information.
- Run the **display ospf [process-id] error [lsa | interface interface-type interface-number]** command in any view to check information about OSPF error information.
- Run the **display ospf [process-id] vlink** command in any view to check information about OSPF virtual links.
- Run the **display ospf [process-id] interface [all | no-peer | interface-type interface-number] [verbose]** command in any view to check information about OSPF interfaces.
- Run the **display ospf [process-id] routing** command in any view to check information about the OSPF routing table.
- Run the **display ospf [process-id] topology [area area-id] [statistics | verbose]** command in any view to check information about the topology calculated for OSPF routes.
- Run the **display ospf [process-id] spf-statistics [verbose]** command in any view to check information about route calculation statistics in OSPF processes.
- Run the **display ospf [process-id] request-queue [interface-type interface-number] [neighbor-id]** command in any view to check information about OSPF request list.
- Run the **display ospf [process-id] retrans-queue [interface-type interface-number] [neighbor-id]** command in any view to check information about OSPF retransmission list.
- Run the **display ospf [process-id] statistics updated-lsa [originate-router advertising-router-id | history]** command in any view to check information about the frequent updates of the LSAs that the LSDB receives
- Run the **display ospf [process-id] statistics maxage-lsa** command to check information about router LSAs that have reached the aging time.
- Run the **display ospf [process-id] router-id conflict** command in any view to check information about router ID conflicts (if any).

----End

1.1.4.2.4 Configuring OSPF on the NBMA or P2MP Network

This section describes how to configure OSPF and modify attributes on the NBMA or point-to-multipoint (P2MP) network to flexibly construct the OSPF network.

Usage Scenario

OSPF classifies networks into four types according to link layer protocols, as shown in [Table 1-48](#).



This section describes only the OSPF configurations that are different on the NBMA network and P2MP network. The OSPF configurations not provided here are applicable to the four types of networks.

Table 1-48 Network types supported by OSPF

Network Type	Characteristic	Default Configuration
Broadcast	On such a type of network, Hello, LSU, and LSAck packets are multicast; DD and LSR packets are unicast.	If the link layer protocol is Ethernet or Fiber Distributed Data Interface (FDDI), OSPF regards the network as a broadcast network by default.
Non-broadcast multiple access (NBMA)	On such a type of network, Hello, DD, LSR, LSU, and LSAck packets are unicast. The NBMA network must be fully meshed. Any two routers on the NBMA network must be directly reachable.	-
Point-to-point (P2P)	On such a type of network, Hello, DD, LSR, LSU, and LSAck packets are multicast.	If the link layer protocol is PPP, Link Access Procedure Balanced (LAPB), OSPF regards the network as a P2P network by default.
Point-to-multipoint (P2MP)	On such a type of network, Hello packets are multicast; DD, LSR, LSU, and LSAck packets are unicast. The mask lengths of the devices on the P2MP network must be the same.	OSPF does not regard a network as a P2MP network by default regardless of any link layer protocol. A P2MP network is forcibly changed from the network of another type.

As shown in **Table 1-48**, OSPF sends packets in different manners on networks of different types. Therefore, the difference between OSPF configurations on the networks lies in the packet sending configurations.

Pre-configuration Tasks

Before configuring session parameters for the OSPF neighbor or adjacency relationship, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic OSPF functions.**

Configuring Network Types for OSPF Interfaces

OSPF classifies networks into four types based on the types of link layer protocols. You can configure the network type for an OSPF interface to forcibly change its original network type.

Context

Network types of OSPF interfaces on both ends of a link must be the same. Otherwise, the two interfaces cannot establish an OSPF neighbor relationship. By default, the network type of an interface is determined by the physical interface. The network type of Ethernet interfaces is **broadcast**.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

Step 3 Run **ospf network-type { broadcast | nbma | p2mp | p2p }**

A network type is configured for the OSPF interface.

After a new network type is configured for an interface, the original network type of the interface is replaced.

Configure a network type for the interface as required. For example:

- If the network type of the interface is broadcast but there are routers that do not support multicast addresses on the broadcast network, change the network type of the interface to NBMA.
- If the network type of the interface is NBMA and the network is fully meshed (any two routers are directly reachable), you can change the network type of the interface to broadcast without the need to configure neighboring routers.
- If the network type of the interface is NBMA but the network is not fully meshed, change the network type of the interface to P2MP. In this manner, two routers that are not directly reachable can exchange routing information through a router that is directly reachable to both routers. After the network type of the interface is changed to P2MP, there is no need to manually configure neighboring routers.
- If only two routers run OSPF on the same network segment, changing the network type of the interface to P2MP is recommended.

NOTE

OSPF does not support the configuration on a null interface.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring NBMA Network Attributes

To implement OSPF functions, configure NBMA network attributes.

Procedure

Step 1 (Optional) Set the network type to NBMA.

The NBMA network must be fully meshed. Any two routers on the NBMA network must be directly reachable. In most cases, however, this requirement cannot be met. To resolve this problem, change the network type to P2MP. For details, see [Configuring Network Types for OSPF Interfaces](#).

1. Run **system-view**

The system view is displayed.

2. Run **interface interface-type interface-number**

The interface view is displayed.

3. Run **ospf network-type nbma**

The network type of the OSPF interface is set to NBMA.

4. Run **commit**

The configuration is committed.

Step 2 (Optional) Run **ospf timer poll interval**

The interval at which Hello packets for polling are sent by an NBMA interface is set.

On the NBMA network, after the neighbor relationship becomes invalid, the router sends Hello packets at an interval defined in the polling mechanism.

Step 3 (Optional) Run **ospf dr-priority priovalue**

A priority is configured for the interface to compete for the DR.

The priority of an interface determines whether the interface is qualified to be a DR. The interface with the highest priority is elected as the DR. If the priority of an interface on a device is 0, the device cannot be elected as a DR or BDR. On a broadcast or an NBMA network, you can set the priority of an interface to control the DR or BDR selection. When the DR and BDR are elected on a network segment, they send DD packets to all neighboring nodes and set up adjacencies with all neighboring nodes.

Step 4 (Optional) Run **ospf timer wait interval**

The wait time is configured for the interface.

If no Backup Seen event is received within the *interval*, the DR election starts. Setting a proper value for the wait timer can slow down changes of the DR and BDR on the network, reducing network flapping. When setting the wait timer, note the following points:

- The wait timer takes effect only on broadcast and NBMA interfaces.
- The value of the wait timer cannot be greater than the value of the dead timer.

Step 5 Configure a neighboring router on the NBMA network.

The interface with the network type of NBMA cannot broadcast Hello packets to discover neighboring routers. Therefore, the IP address of a neighboring router must be configured on the process and whether the neighboring router can participate in DR election must be determined on the process.

1. Run **quit**

Exit the interface view.

2. Run **ospf [process-id]**

The OSPF view is displayed.

3. Run **peer ip-address [dr-priority priority]**

A neighbor is configured on the NBMA network.

4. Run **commit**

The configuration is committed.

Step 6 Run **commit**

The configuration is committed.

----End

Configuring P2MP Network Attributes

This section describes how to configure P2MP network attributes to implement OSPF functions.

Procedure

Step 1 Disable the check on the network mask.

The OSPF neighbor relationship cannot be established between the devices with different mask lengths on the P2MP network. After OSPF is disabled from checking the network mask, the OSPF neighbor relationship can be properly established.

1. Run **system-view**

The system view is displayed.

2. Run **interface interface-type interface-number**

The interface view is displayed.

3. Run **ospf network-type p2mp**

The network type of the OSPF interface is configured.

A P2MP network is forcibly changed from another other type of network. For details, see [Configuring Network Types for OSPF Interfaces](#).

4. Run **ospf p2mp-mask-ignore**

OSPF is disabled from checking the network mask on the P2MP network.

5. Run **commit**

The configuration is committed.

Step 2 (Optional) Configure the device to filter the LSA packets to be sent.

When multiple links exist between two devices, you can configure the local device to filter the LSA packets to be sent. This can reduce unnecessary LSA retransmission attempts and save bandwidth resources.

1. Run **system-view**

The system view is displayed.

2. Run **ospf [process-id]**

The OSPF view is displayed.

3. Based on the basic ACL:

- a. Run **filter-lsa-out peer ip-address { all | { summary [acl { acl-number | acl-name }] | ase [acl { acl-number | acl-name }] | nssa [acl { acl-number | acl-name }] } * }**

The local device is configured to filter the LSA packets to be sent on the P2MP network.

b. Run **quit**

Return to the system view.

- c. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**

The ACL view is displayed.

- d. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

4. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After OSPF attributes are set on networks of different types, you can check OSPF neighbor and interface information.

Prerequisites

OSPF attributes have been configured on networks of different types.

Procedure

- Run the **display ospf [process-id] interface [all | no-peer | interface-type interface-number] [verbose]** command to check information about OSPF interfaces.
- Run the **display ospf [process-id] peer** command to check information about OSPF neighbors.
- Run the **display ospf brief** command to check the interval at which Hello packets are sent on an NBMA network.

----End

1.1.4.2.5 Adjusting OSPF Route Selection

By adjusting OSPF route selection, you can enable OSPF to meet the requirements of complex networks.

Usage Scenario

In real world situations, you can configure an OSPF route selection rule by setting OSPF route attributes to meet the requirements of complex networks.

- Set the cost of an interface. The link connected to the interface with a smaller cost value preferentially transmits routing information.
- Configure equal-cost routes to implement load balancing.
- Configure a stub router during the maintenance operations such as upgrade to ensure stable data transmission through key routes.
- Suppress interfaces from sending or receiving packets to help select the optimal route.

- Configuring an OSPF interface to automatically adjust the link cost based on link quality that facilitates route selection control and improves network reliability.

Pre-configuration Tasks

Before adjusting OSPF route selection, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic OSPF functions.**

Setting the Interface Cost

You can adjust and optimize route selection by setting the OSPF interface cost.

Context

After the OSPF interface costs are set, the interface with a smaller cost value preferentially transmits routing information. This helps select the optimal route.

The OSPF interface cost can be set or calculated based on the interface bandwidth.

External factors may affect the physical bandwidth of links and change the physical bandwidth of interfaces, which in turn affects network performance. To address this problem, you can run the **bandwidth bandwidth** command in the interface view to set configuration bandwidth for the interface, and then run the **bandwidth-config enable** command to enable the device to calculate the cost for the OSPF interface based on the configuration bandwidth of the interface.

Procedure

- Manually configure a cost for an OSPF interface.
 - Run **system-view**
The system view is displayed.
 - Run **interface interface-type interface-number**
The interface view is displayed.
 - Run **ospf cost value**
A cost is configured for the OSPF interface.
 - Run **commit**
The configuration is committed.
- Configure bandwidth-based automatic OSPF interface cost calculation.
The calculation formula is as follows: **Interface cost = Bandwidth reference value/Interface bandwidth**. The integer of the calculation result is used as the cost of the interface. If the result is less than 1, the cost is 1.

 NOTE

Perform the following operations as required:

- To use the bandwidth reference value to determine the interface cost, run the **bandwidth-reference** command.
- To use the interface bandwidth to determine the interface cost, run the **bandwidth** command to set configuration bandwidth for an interface, and then run the **bandwidth-config enable** command to enable the device to calculate the cost for the OSPF interface based on the configuration bandwidth of the interface.
- To use both the bandwidth reference value and interface bandwidth to determine the interface cost, run the preceding three commands.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **bandwidth bandwidth**

Configuration bandwidth is configured for the interface.

d. Run **quit**

Exit the interface view.

e. Run **ospf [process-id]**

The OSPF view is displayed.

f. Run **bandwidth-reference value**

A bandwidth reference value is set.

g. Run **bandwidth-config enable**

Configuration bandwidth of each OSPF interface is allowed to participate in cost calculation for the interface.

 NOTE

- If the **bandwidth** command is not run, the cost of an OSPF interface is calculated based on the physical bandwidth of the interface.
- If the **bandwidth-config enable** command is not run, the cost of an OSPF interface is calculated based on the physical bandwidth of the interface.

h. Run **commit**

The configuration is committed.

- Associate the remaining bandwidth of an OSPF interface with the link cost.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Associate the remaining bandwidth of an OSPF interface with the link cost.

Perform the following operations as needed:

- To associate the remaining bandwidth of an OSPF interface with the link cost, run the **ospf cost value { higher-bandwidth higher-bandwidth-value cost better-cost-value | lower-bandwidth lower-bandwidth-value cost worse-cost-value } *** command.
- To associate the remaining bandwidth of an OSPF multi-area adjacency interface with the link cost, run the **ospf cost costvalue { higher-bandwidth higher-bandwidth-value cost better-cost-value | lower-bandwidth lower-bandwidth-value cost worse-cost-value } * multi-area { area-id-integer | area-id-ipv4 }** command.

d. Run **commit**

The configuration is committed.

----End

Configuring Equal-Cost Routes

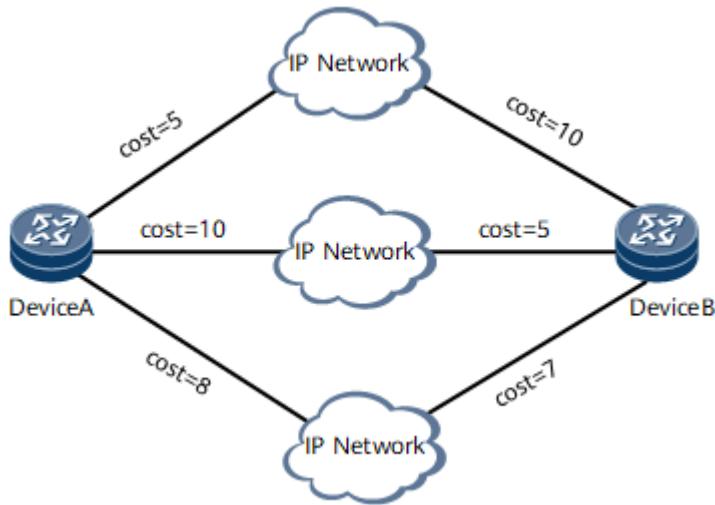
You can set the number of OSPF equal-cost routes and route preference to implement load balancing and adjust route selection.

Context

If the destinations and costs of the multiple routes discovered by one routing protocol are the same, load balancing can be implemented among the routes.

As shown in [Figure 1-120](#), three routes between Device A and Device B that run OSPF have the same costs. The three routes are equal-cost routes for load balancing.

Figure 1-120 Networking diagram of equal-cost routes



Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **maximum load-balancing number**

The maximum number of equal-cost routes is set.

If the number of equal-cost routes is greater than *number* specified in the **maximum load-balancing** command, routes are selected for load balancing based on the following criteria:

1. Route preference: Routes with smaller preference values are selected for load balancing. For details about route preference configuration, see [Step 4](#).
2. Interface index: If routes have the same preference, those with greater interface index values are selected for load balancing.
3. Next hop IP address: If routes have the same preference and interface index, those with larger IP addresses are selected for load balancing.

Step 4 (Optional) Run either of the following commands:

- To configure a priority for an equal-cost route, run the **nexthop ip-address weight value** command.

Among equal-cost OSPF routes, those with smaller priority values configured using the **nexthop** command are selected for load balancing.

 - *ip-address* specifies the next hop address of the equal-cost route.
 - *value* specifies the weight of the next hop.
- To configure priorities for the routes with a TE tunnel interface or an IPv4 interface as the outbound interface for route selection if both an IGP-Shortcut-enabled TE tunnel and IP link are available, run the **ecmp prefer { te-tunnel | intact }** command.

Step 5 Run **commit**

The configuration is committed.

----End

Setting the convergence priority for OSPF routes

You can adjust and optimize route selection by setting the convergence priority for OSPF routes.

Context

You can set a convergence priority for the OSPF routes matching the specified IP prefix list. The configuration takes effect on the public network only.

OSPF route calculation, link-state advertisement (LSA) flooding, and LSDB synchronization can be implemented according to the configured priority. Therefore, route convergence can be controlled.

When an LSA meets multiple priorities, the highest priority takes effect.

OSPF calculates LSAs in the sequence of intra-area routes, inter-area routes, and AS external routes and calculates the three types of routes separately according to the specified route calculation priorities. Convergence priorities are critical, high,

medium, and low. To speed up the processing of LSAs with the higher priority, during LSA flooding, the LSAs need to be placed into the corresponding critical, high, medium, and low queues according to priorities.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip ip-prefix ip-prefix-name [index index-number] { permit | deny } ipv4-address mask-length [match-network] [greater-equal greater-equal-value] [less-equal less-equal-value] [description text]**

An IP prefix list is configured.

Step 3 Run **ospf [process-id]**

The OSPF view is displayed.

Step 4 Run **prefix-priority { critical | high | medium } ip-prefix ip-prefix-name**

The convergence priority for OSPF routes is set.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring a Stub Router

To ensure that a route is not interrupted during flapping-triggering maintenance operations such as upgrade, you can configure a router as a stub router to allow traffic to bypass the route on the stub router.

Context

After a stub router is configured, the route on the stub router will not be preferentially selected. After the route cost is set to the maximum value 65535, traffic generally bypasses the router. This ensures an uninterrupted route on the router during maintenance operations such as upgrade.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **stub-router [[on-startup [interval]] | [include-stub] | [external-lsa [external-lsa-metric]] | [summary-lsa [summary-lsa-metric]]] ***

A stub router is configured.

 NOTE

The stub router configured in this manner is irrelevant to the router in the stub area.

Step 4 (Optional) Run **maximum-link-cost cost**

The maximum cost of OSPF is set.

Step 5 Run **commit**

The configuration is committed.

----End

Suppressing an Interface from Receiving and Sending OSPF Packets

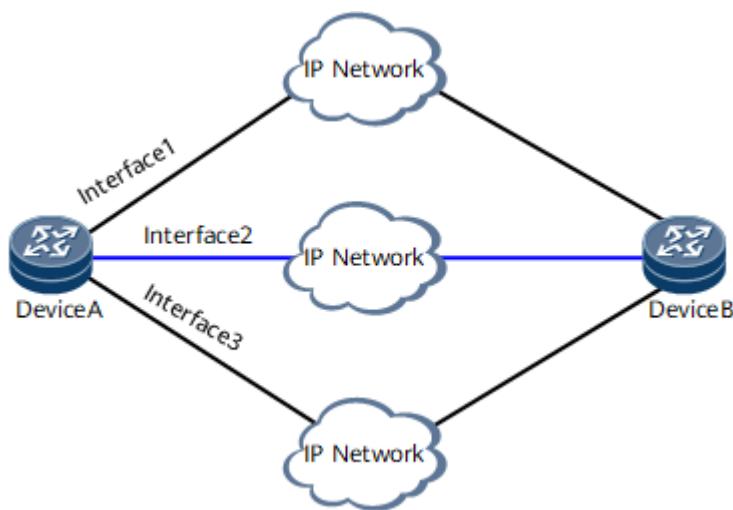
After an interface is suppressed from receiving and sending OSPF packets, routing information can bypass a specific router and the local router can reject routing information advertised by another router.

Context

Suppressing an interface from receiving and sending OSPF packets helps routing information to bypass a specific router and enables the local router to reject routing information advertised by another router. This ensures that an optimal route is provided.

For example, there are three routes between DeviceA and DeviceB, as shown in [Figure 1-121](#). To allow the route with the outbound interface being Interface 2 to be preferred, you can suppress Interface 1 and Interface 3 from receiving and sending OSPF packets.

Figure 1-121 Network diagram of suppressing the interfaces from sending and receiving OSPF packets



Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **silent-interface { all | interface-type interface-number }**

An interface is suppressed from receiving and sending OSPF packets.

The same interface in different processes can be suppressed from sending and receiving OSPF packets, but the **silent-interface** command is valid only for the OSPF interface in the local process.

After an OSPF interface is configured to be in the silent state, the interface can still advertise its direct routes. Hello packets on the interface, however, cannot be forwarded. Therefore, no neighbor relationship can be established on the interface. This can enhance the networking adaptability of OSPF and reduce system resource consumption.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring an OSPF Interface to Automatically Adjust the Link Cost

Configuring an OSPF interface to automatically adjust the link cost based on link quality facilitates route selection control and improves network reliability.

Context

A bit error refers to the deviation between a bit that is sent and the bit that is received. The bit error rate (BER) refers to the number of bit errors divided by the total number of bits transferred during a studied time interval. During data transmission, a high BER will degrade or even interrupt services in extreme cases.

To prevent this problem, configure OSPF interfaces to automatically adjust link costs based on link quality so that unreliable links are not used by the optimal routes.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

Step 3 Run **ospf enable [process-id] area area-id**

OSPF is enabled on the interface.

Step 4 Run **ospf link-quality low incr-cost { cost | max-reachable }**

The OSPF interface is configured to automatically adjust the link cost based on link quality.

 NOTE

The *cost* parameter specifies the link cost adjustment value. After this parameter is specified:

- If the link quality changes to **low**, the link cost equals the original link cost plus the adjustment value. The maximum link cost allowed is 65535.
- If the link quality recovers from **low**, the original link cost applies.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring a Fallback Cost for an Eth-Trunk Interface

A fallback cost of an Eth-Trunk interface helps adjust route selection dynamically.

Context

After an Eth-trunk member interface goes down, the OSPF cost value is automatically adjusted. When an Eth-trunk member interface becomes invalid, the remaining bandwidth may fail to meet user requirements, causing service loss. In this fault scenario, the cost of the Eth-Trunk can be dynamically adjusted to a larger value so that traffic is transmitted through other paths. When the interface bandwidth is less than the fallback bandwidth threshold, the device changes the interface cost to the configured fallback cost in time so that a better transmission path is selected. When the bandwidth of an Eth-Trunk interface is greater than or equal to the configured fallback bandwidth threshold, cost-fallback does not take effect.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface eth-trunk *trunk-id***

The Eth-Trunk interface view is displayed.

Step 3 Run **ospf cost-fallback *fallbackcost* *threshold* *fallbackbw***

A fallback cost is configured for an Eth-Trunk interface.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After OSPF route selection is adjusted, you can check OSPF routing table and interface information.

Prerequisites

OSPF route selection has been adjusted.

Procedure

- Run the **display ospf [process-id] interface [all | no-peer | interface-type interface-number] [verbose]** command to check information about OSPF interfaces.
- Run the **display ospf [process-id] routing** command to check information about the OSPF routing table.
- Run the **display ospf [process-id] ecmp-group** command to check information about OSPF ECMP groups

----End

1.1.4.2.6 Controlling OSPF Routing Information

You can control the advertising and receiving of OSPF routing information and import routes of other protocols.

Pre-configuration Tasks

Before controlling OSPF routing information, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPF functions](#).

Configuring OSPF to Import External Routes

Importing the routes discovered by other routing protocols can enrich OSPF routing information.

Context

When a device on an OSPF network needs to access a device running a non-OSPF routing protocol, the device needs to import the routes of the non-OSPF routing protocol into the OSPF network.

OSPF provides loop-free intra-area routes and inter-area routes; however, OSPF cannot prevent external routing loops. Therefore, exercise caution when configuring OSPF to import external routes.

Perform the following operations on the router that functions as the ASBR running OSPF:

NOTICE

OSPF and other dynamic routing protocols such as IS-IS and BGP often import routes from each other. If no routing policy is configured or a routing policy is incorrectly configured on a device where IS-IS, OSPF, and BGP import routes from each other, a Layer 3 routing loop may occur due to a route selection result change. As a result, services are compromised. For details about the cause of the loop, see [Routing Loop Detection for Routes Imported to OSPF](#).

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ospf [process-id]

The OSPF view is displayed.

Step 3 Run import-route { bgp [permit-ibgp] | direct | rip [process-id-rip] | static | unr | isis [process-id-isis] | ospf [process-id-ospf] } [{ inherit-cost | cost cost } | tag tag | type type | route-policy route-policy-name | route-filter route-filter-name | no-sid] *

The device is configured to import routes from another protocol.

Step 4 (Optional) Run default { cost { cost | inherit-metric } | tag tag | type type } *

The default values of parameters (the cost, number of routes, tag, and type) are set for imported routes.

When OSPF imports external routes, you can set default values for some additional parameters, such as the cost, number of routes to be imported, route tag, and route type. The route tag is used to identify the protocol-related information. For example, it can be used to differentiate AS numbers carried in BGP routes imported by OSPF.

NOTE

You can run one of the following commands to set the cost of the imported route. The following commands are listed in descending order of priorities.

- Run the **apply cost** command to set the cost of a route.
- Run the **import-route** command to set the cost of the imported route.
- Run the **default** command to set the default cost of the imported route.

Step 5 (Optional) Run any of the following commands as required:

- Based on a basic ACL:

- a. Run **filter-policy { acl-number| acl-name acl-name } export [direct | static | unr | bgp | { rip | isis | ospf } [process-id]]**

The device is configured to filter the routes imported in step [Step 3](#) so that only the matched routes are advertised.

- b. Run **quit**

Return to the system view.

- c. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**

The ACL view is displayed.

- d. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

e. Run **ospf [process-id]**

The OSPF view is displayed.

- Based on an IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name export [direct | static | unr | bgp | { rip | isis | ospf } [process-id]]**

The device is configured to filter the routes imported in step **Step 3** so that only the matched routes are advertised.

OSPF filters the imported routes. OSPF uses Type 5 LSAs to carry routes that meet the filtering conditions and advertises these Type 5 LSAs.

You can specify the parameter **protocol [process-id]** to filter the routes of a certain routing protocol or a certain OSPF process. If **protocol [process-id]** is not specified, OSPF filters all imported routes.

The **import-route** command cannot be used to import external default routes.

Step 6 Run **import-route limit limit-number [threshold-alarm { upper-limit upper-limit-value | lower-limit lower-limit-value } *]**

The maximum number of LSAs generated when OSPF imports routes is set.

If OSPF imports a large number of external routes and advertises them to a device with a small routing table capacity, the device may restart unexpectedly. To address this problem, set a limit on the number of LSAs generated when OSPF imports external routes. You can check the overload status based on the value of the **Current status** field in the **display ospf brief** command output.

- Normal: The number is less than or equal to the lower alarm threshold.
- Approach limit: The number is approaching (reaching or exceeding 90% of) the upper alarm threshold.
- Exceed limit: The number has reached or exceeded the limit.

Ensure that *upper-limit-value* is greater than or equal to *lower-limit-value*.

Step 7 Run **commit**

The configuration is committed.

----End

Configuring OSPF to Import a Default Route

The default route is widely applied on the OSPF network to reduce routing entries in the routing table and filter specific routing information.

Context

On the area border and AS border of an OSPF network generally reside multiple routers for next-hop backup or traffic load balancing. A default route can be configured to reduce routing entries and improve resource usage on the OSPF network.

OSPF default routes are generally applied to the following scenarios:

1. An ABR in an area advertises Type 3 LSAs carrying the default route within the area. routers in the area use the received default route to forward inter-area packets.
2. An ASBR in an AS advertises Type 5 or Type 7 LSAs carrying the default route within the AS. routers in the AS use the received default route to forward AS external packets.

When no exactly matched route is discovered, the router can forward packets through the default route.

The priorities of the default routes corresponding to Type 3 LSAs are higher than those of the routes corresponding to Type 5 or Type 7 LSAs.

The advertising mode of OSPF default routes is determined by the type of the area to which the default routes are imported, as shown in [Table 1-49](#).

Table 1-49 Default route advertising mode

Area Type	Generated By	Advertised By	LSA Type	Flooding Area
Common area	The default-route-advertise command	ASBR	Type 5 LSA	Common area
Stub area	Automatically	ABR	Type 3 LSA	Stub area
NSSA	The nssa [default-route-advertise] command	ASBR	Type 7 LSA	NSSA
	Automatically	ABR	Type 3 LSA	NSSA
Totally NSSA	Automatically	ABR	Type 3 LSA	NSSA

Perform the following steps on the ASBR running OSPF.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **default-route-advertise [[always | permit-calculate-other] | cost cost | type type | { route-policy route-policy-name | route-filter route-filter-name } | distribute-delay delay-time | permit-preference-less-than preference-val] ***

The default route is imported to the OSPF process.



To prevent loops, you are advised to use the **permit-preference-less-than** parameter to prevent the import of activated default routes with low priorities. This parameter is used only when the **always** parameter is not specified.

Configure the default route for the NSSA. For details, see [1.1.4.2.15 Configuring an NSSA](#).

Step 4 Run **commit**

The configuration is committed.

----End

Configuring Route Summarization

When a large-scale OSPF network is deployed, you can configure route summarization to reduce routing entries.

Context

Route summarization on a large-scale OSPF network reduces routing entries, minimizes system resource consumption, and maintains system performance. In addition, if a specific link frequently alternates between Up and Down, the links not involved in the route summarization will not be affected, which prevents route flapping and improves network stability.

Procedure

- Configure ABR route summarization.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **ospf [process-id]**

The OSPF view is displayed.
 - c. Run **area area-id**

The OSPF area view is displayed.
 - d. Run **abr-summary ip-address mask [[advertise | [cost { cost-value | inherit-minimum }] | [hold-max-cost hours] | [generate-null0-route]] * | [not-advertise | [cost { cost-value | inherit-minimum }] | [hold-max-cost hours]] * | [generate-null0-route | [advertise] | [cost { cost-value | inherit-minimum }] | [hold-max-cost hours]] *]**

OSPF ABR route summarization is configured.
 - e. Run **commit**

The configuration is committed.
- Configure ASBR route summarization.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **ospf [process-id]**

The OSPF view is displayed.
 - c. Run **asbr-summary ip-address mask [[not-advertise | generate-null0-route] | tag tag-value | cost cost-value | distribute-delay interval] *]**

OSPF route summarization is configured on the ASBR.
 - d. Run **commit**

The configuration is committed.

 NOTE

After route summarization is configured, the routing table of the local OSPF device remains unchanged. The routing table on an OSPF neighbor, however, contains only one summary route and no specific routes. This summary route stays in the routing table until all the summarized specific routes on the network are withdrawn.

----End

Configuring OSPF to Filter LSAs in an Area

Filtering LSAs in an area can prevent unnecessary LSA transmission. This reduces the size of the LSDB on the neighboring router and speeds up network convergence.

Context

After filtering conditions are set for the incoming or outgoing Type 3 LSAs (Summary LSAs) in an area, only the Type 3 LSAs that meet the filtering conditions can be received or advertised.

This function is applicable only to the ABR.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **area area-id**

The OSPF area view is displayed.

Step 4 Filter incoming or outgoing Type 3 LSAs in the area.

- Filter outgoing Type 3 LSAs in the area. Run any of the following commands as required:

- Based on the basic ACL:

- i. Run **quit**

Return to the system view.

- ii. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**

The ACL view is displayed.

- iii. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

iv. Run **ospf [process-id]**

The OSPF view is displayed.

v. Run **area area-id**

The OSPF area view is displayed.

vi. Run **filter { acl-number | acl-name acl-name } export**

The outgoing Type 3 LSAs of the local area are filtered based on the basic ACL.

- Based on the IP prefix list:

Run **filter ip-prefix ip-prefix-name export**

Outgoing Type 3 LSAs in the area are filtered based on the IP prefix list.

- Based on the route-policy:

Run **filter route-policy route-policy-name export**

Outgoing Type 3 LSAs in the area are filtered based on the route-policy.

- Based on a route-filter:

Run **filter route-filter route-filter-name export**

Outgoing Type 3 LSAs in the area are filtered based on the route-filter.

- Filter incoming Type 3 LSAs in the area. Run any of the following commands as required:

- Based on the basic ACL:
 - i. Run **quit**
Return to the system view.
 - ii. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**
The ACL view is displayed.
 - iii. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***
An ACL rule is configured.
When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.
When a filter-policy of a routing protocol is used to filter routes:
 - o If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
 - o If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
 - o If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
 - o If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
 - o Routes can be filtered using a blacklist or whitelist:
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.
- iv. Run **ospf [process-id]**
The OSPF view is displayed.
- v. Run **area area-id**
The OSPF area view is displayed.
- vi. Run **filter { acl-number | acl-name acl-name } import [include-abr-summary]**

Incoming Type 3 LSAs in the area are filtered based on the basic ACL.

- Based on the IP prefix list:

Run **filter ip-prefix ip-prefix-name import [include-abr-summary]**

Incoming Type 3 LSAs in the area are filtered based on the IP prefix list.

- Based on the route-policy:

Run **filter route-policy route-policy-name import [include-abr-summary]**

Incoming Type 3 LSAs in the area are filtered based on the route-policy.

- Based on the route-filter:

Run **filter route-filter route-filter-name import [include-abr-summary]**

Incoming Type 3 LSAs in the area are filtered based on the route-filter.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring OSPF to Filter LSAs to Be Sent

Filtering the LSAs to be sent on the local router can prevent unnecessary LSA transmission. This reduces the size of the LSDB on the neighboring device and speeds up network convergence.

Context

When multiple links exist between two devices, you can configure the local device to filter the LSAs to be sent. This prevents unnecessary LSA transmission and saves bandwidth resources.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

Step 3 Based on the basic ACL:

1. Run **quit**

Return to the system view.

2. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**

The ACL view is displayed.

3. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

4. Run **quit**

Return to the system view.

5. Run **interface interface-type interface-number**

The interface view is displayed.

6. Run **ospf filter-lsa-out { all | { summary [acl { acl-number| acl-name }] | ase [acl { acl-number| acl-name }] | nssa [acl { acl-number| acl-name }] } * }**

The OSPF interface is enabled to filter outgoing LSAs.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring OSPF to Filter Received Routes

After a filtering policy is configured for the OSPF routes that need to be delivered to the routing management module, only the routes that match the policy will be added to the routing table.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Perform any of the following operations as required:

- Based on the basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**

The ACL view is displayed.

- c. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- d. Run **ospf** [*process-id*]
The OSPF view is displayed.
- e. Run **filter-policy** { *acl-number* | **acl-name** *acl-name* } **import**
An import policy is configured to filter received routes.
- Based on the IP prefix list:
Run **filter-policy ip-prefix** *ip-prefix-name* **import**
An import policy is configured to filter received routes.
- Based on the route-policy:
Run **filter-policy route-policy** *route-policy-name* [**secondary**] **import**
An import policy is configured to filter received routes.
- Based on the route-filter:
Run **filter-policy route-filter** *route-filter-name* [**secondary**] **import**
An import policy is configured to filter received routes.

OSPF is a link-state dynamic routing protocol, with routing information carried in the link status advertisement (LSA). Therefore, the **filter-policy import** command cannot be used to filter the advertised or received LSAs. The **filter-policy import** command is used to filter the routes calculated by OSPF. Only the routes that match the filtering rules are added to the routing table and can be advertised. Routes that do not match the filtering rules can be added to the OSPF routing table but not to the routing information base (RIB) and cannot be advertised.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring OSPF to Filter the Routes to Be Advertised

After a filtering policy is configured for routes imported by OSPF, only the routes that match the policy can be advertised.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf** [*process-id*]

The OSPF view is displayed.

Step 3 Set the conditions to filter routes to be advertised.

Run any of the following commands as required:

- Based on a basic ACL:
 - a. Run **quit**
Return to the system view.
 - b. Run **acl** { **name** *basic-acl-name* { **basic** | [**basic**] **number** *basic-acl-number* } | [**number**] *basic-acl-number* } [**match-order** { **config** | **auto** }]

The basic ACL view is displayed.

- c. Run **rule** [*rule-id*] [**name** *rule-name*] { **deny** | **permit** } [**fragment-type** { **fragment** | **non-fragment** | **non-subseq** | **fragment-subseq** | **fragment-spe-first** }] **source** { *source-ip-address* { *source-wildcard* | **0** | *src-netmask* } | **any** } | **time-range** *time-name* | **vpn-instance** *vpn-instance-name*] *

A rule is configured for the ACL.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- d. Run **ospf** [*process-id*]

The OSPF view is displayed.

- e. Run **filter-policy** { *acl-number* | **acl-name** *acl-name* } **export** [**direct** | **static** | **unr** | **bgp**] { **rip** | **isis** | **ospf** } [*process-id*]]

The device is configured to filter the imported routes to be advertised.

- Based on an IP prefix list:

Run **filter-policy ip-prefix** *ip-prefix-name* **export** [**direct** | **static** | **unr** | **bgp** | { **rip** | **isis** | **ospf** } [*process-id*]]

The device is configured to filter the imported routes to be advertised.

- Based on a route-policy:

Run **filter-policy route-policy-name export [direct | static | unr | bgp | { rip | isis | ospf } [process-id]]**

The device is configured to filter the imported routes to be advertised.

Step 4 Run commit

The configuration is committed.

----End

(Optional) Configuring OSPF to Discard Specified LSAs

You can configure a device to discard specified LSAs in an OSPF process.

Context

OSPF can be configured to discard specified LSAs in the following scenarios:

1. When devices on the entire network restart repeatedly due to abnormal LSAs and you have located the LSA that causes protocol restarts, you can configure this function as a last resort to prevent the device from restarting continuously. However, if this function is incorrectly configured, routing loops may occur.
2. If an LSA is identified as an attack packet, which is not supposed to appear in the local area, and has caused serious problems, such as device restarts, you can configure this function to filter out the LSA under the condition that the attack source cannot be located temporarily and that the LSA does not affect topology path computation.
3. If an LSA is identified as an attack packet, which is not supposed to appear in the local area, and it affects topology path computation and has caused serious problems, such as network-wide device restarts, you can configure this function on each device to discard the LSA to prevent it from participating in network-wide calculation.

NOTE

To filter out the LSA that affects topology path computation, you must ensure that it is removed from all the LSDBs on the entire network. Otherwise, routing loops may occur.

4. If an LSA is identified as an unreachable residual LSA and the device that advertised the LSA becomes permanently unreachable, you can configure this function to filter out the LSA upon reception under the condition that the LSA does not affect topology path computation.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ospf [process-id]

The OSPF view is displayed.

Step 3 Run ignore-receive-lsa advertise-router adv-rtr-id [lsa-type type-value [area { area-id | area-idipv4 }]] | link-state-id ls-id] *

The device is configured to discard LSAs of a specified type.

 NOTE

If this command is incorrectly configured, services cannot be restored even if the **undo ignore-receive-lsa advertise-router adv-rtr-id [lsa-type type-value [area { area-id | area-idipv4 }] | link-state-id ls-id] *** command is run. In this case, you may need to reset the process or neighbor to restore services.

You are not advised to run this command to filter out the LSAs that exist on the network as running this command may filter out normal LSAs.

As an attack LSA can have any key, it is difficult to defend against the LSA using this command. Therefore, you are advised to directly isolate the attack source.

This command cannot be used to defend against attacks as it goes against protocol processing rules and affects services. Therefore, exercise caution when running this command.

If the fault is caused by a bug, you are advised to run this command temporarily. After the patch is installed, run the **undo ignore-receive-lsa advertise-router adv-rtr-id [lsa-type type-value [area { area-id | area-idipv4 }] | link-state-id ls-id] *** command immediately and check whether services are affected. If services are affected, re-establish all neighbor relationships to restore services.

Step 4 Run commit

The configuration is committed.

----End

Configuring the Maximum Number of External Routes Supported by the OSPF LSDB

You can set the maximum number of external routes in the LSDB to keep a proper number of external routes.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run ospf [process-id]

The OSPF view is displayed.

Step 3 Run lsdb-overflow-limit number

The maximum number of external routes in the LSDB is set.

If the number of external routes imported by OSPF exceeds the maximum number allowed by the LSDB, the device deletes self-generated non-default external routes to ensure proper forwarding of the other external routes imported by OSPF.

Step 4 Run commit

The configuration is committed.

----End

Verifying the Configuration of OSPF Routing Information Control

After OSPF routing information is controlled, you can check OSPF LSDB information.

Prerequisites

OSPF routing information has been controlled.

Procedure

- Run the **display ospf [process-id] lsdb** command to check information about the OSPF LSDB.
- Run the **display ospf [process-id] asbr-summary [ip-address mask]** command to check information about OSPF route summarization.

----End

1.1.4.2.7 Adjusting the OSPF Network Convergence Speed

By adjusting OSPF timers, you can implement OSPF network convergence speed.

Pre-configuration Tasks

Before adjusting the OSPF network convergence speed, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPF functions](#).

Setting the Interval at Which Hello Packets Are Sent

You can adjust the value of the Hello timer to change the speed of the OSPF neighbor relationship establishment and change the network convergence speed.

Context

Hello packets are periodically sent between OSPF interfaces to establish and maintain neighbor relationships. The intervals set on the interfaces at both ends must be the same. Otherwise, the OSPF neighbor relationship cannot be established.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The OSPF interface view is displayed.

Step 3 Run **ospf timer hello interval [conservative]**

The interval at which Hello packets are sent is set on the interface.

The **conservative** parameter indicates that the conservative mode is enabled for the neighbor dead timer. If the conservative mode is enabled, the value configured for the dead timer using the **ospf timer dead** command takes effect even when the value is less than 10s.

To speed up OSPF convergence in the case of a link failure, configuring BFD for OSPF is recommended. If the remote end does not support BFD for OSPF or the user does not want to have BFD for OSPF enabled, you are advised to specify **conservative** when running the **ospf timer dead** command. In conservative mode, the value set for the dead timer using the **ospf timer dead** command takes effect even if the value is less than 10 seconds; otherwise, route convergence is performed based on the OSPF neighbor dead timer, which takes a long time and has a great impact on services.

NOTE

This interval must not be less than the time taken to perform an active/standby switchover. Otherwise, an intermittent protocol interruption may occur during a switchover. The default timer value is recommended.

Step 4 Run **commit**

The configuration is committed.

----End

Setting the Delay for Transmitting LSAs on an Interface

This configuration is important for low-speed networks.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The OSPF interface view is displayed.

Step 3 Run the **ospf trans-delay delayvalue** command to set the delay for the interface to transmit LSAs.

The LSAs in the LSDB of the local device age with time (increase by 1 each second), but not during transmission. Therefore, an LSA transmission delay needs to be set before LSAs are sent.

Step 4 Run **commit**

The configuration is committed.

----End

Setting the Dead Time of an OSPF Neighbor

If no Hello packet is received from a neighbor within a dead interval, the neighbor is considered Down.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The OSPF interface view is displayed.

Step 3 Run **ospf timer dead interval**

A dead timer is set for OSPF neighbor relationships.

NOTE

If the dead timer is shorter than 10s, the neighbor relationship may be torn down. To prevent this issue, the dead timer of 10 seconds takes effect if the value of **dead interval** configured for OSPF neighbor relationships is less than 10 seconds. However, if the **ospf timer hello** command is run, the **conservative** parameter is specified to enable the conservative mode for the neighbor dead timer, and the configured dead timer is less than 10s, the system still uses the configured value to determine whether a neighbor is down.

Changing the network type will restore both the Hello timer and dead timer to their default values.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring OSPF Sham Hello

With OSPF sham hello, device can exchange Hello and LSU and LSAck packets to maintain OSPF neighbor relationships, which strengthens the neighbor detection mechanism.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **sham-hello enable**

OSPF sham hello is enabled.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring Smart-discover

After Smart-discover is configured, when the neighbor status changes or the DR or BDR on the multiple-access network (broadcast network or NBMA network)

changes, the local router sends Hello packets to its neighbor immediately without waiting for the Hello timer to expire.

Context

Without Smart-discover, when the neighbor status of the router changes or the DR/BDR on the multiple-access network (broadcast or NBMA network) changes, the router does not send Hello packets to its neighbor until the Hello timer expires; after Smart-discover is configured, the router sends Hello packets to its neighbor immediately without waiting for the Hello timer to expire, which speeds up the neighbor relationship establishment and OSPF network convergence.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The OSPF interface view is displayed.

Step 3 Run **ospf smart-discover**

Smart-discover is configured on the interface.

Step 4 Run **commit**

The configuration is committed.

----End

Setting the Interval at Which LSAs Are Updated

You can set the interval at which LSAs are updated based on network connections and router resources.

Context

OSPF sets the interval for updating LSAs to 5s. This prevents network connections or route flapping from consuming excessive network bandwidth or device resources. On a stable network that requires fast route convergence, you can set the interval for updating LSAs to 0 seconds. In this manner, when the topology or a route changes, the change can be immediately advertised to the network through LSAs. This speeds up route convergence on the network. On an unstable network, routes are calculated frequently, consuming excessive CPU resources. In addition, LSAs that describe the unstable topology are frequently generated and transmitted on the unstable network. Due to the frequent processing of such LSAs, the entire network cannot become stable quickly. You can configure an intelligent timer so that the device can dynamically adjust the interval based on the user configuration and frequency of triggering events (such as route calculation) to quickly stabilize the network.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **lsa-originate-interval { 0 | intelligent-timer max-interval start-interval hold-interval [other-type interval] | other-type interval [intelligent-timer max-interval start-interval hold-interval] }**

The interval at which LSAs are updated is configured.

- **intelligent-timer**: uses the intelligent timer to set the update interval for OSPF Type 1 LSAs (router LSAs) and Type 2 LSAs (network LSAs).
- **max-interval**: specifies the maximum interval at which OSPF LSAs are updated, in milliseconds.
- **start-interval**: specifies the initial interval at which OSPF LSAs are updated, in milliseconds.
- **hold-interval**: specifies the hold interval at which OSPF LSAs are updated, in milliseconds.
- **other-type**: sets the update interval for OSPF Type 3 LSAs (network-summary-LSAs), Type 4 LSAs (ASBR-summary-LSAs), and Type 10 LSAs (opaque LSAs).

The interval for updating LSAs is as follows:

1. The initial interval at which LSAs are updated is specified by *start-interval*.
2. The interval at which LSAs are updated for the nth ($n \geq 2$) time equals *hold-interval* $\times 2^{(n - 2)}$.
3. When the interval specified by *hold-interval* $\times 2^{(n - 2)}$ reaches the maximum interval specified by *max-interval*, OSPF updates LSAs at the maximum interval for three consecutive times. Then, OSPF updates LSAs at the initial interval specified by *start-interval*.

Step 4 (Optional) Run **lsa-originate-interval suppress-flapping suppress-interval [threshold threshold]**

The suppression period that takes effect when sent OSPF LSAs flap is configured.

If no flapping occurs in sent OSPF LSAs, you can run the **lsa-originate-interval** command to prevent frequent LSA sending by setting a proper interval at which LSAs are sent. If sent OSPF LSAs flap, you can run the **lsa-originate-interval suppress-flapping** command to set a flapping suppression period. This minimizes the impact of frequent LSA flapping on services. If both of them are configured, the device uses the larger value as the flapping suppression time.

Step 5 Run **commit**

The configuration is committed.

----End

Setting the Interval at Which LSAs Are Received

You can set the interval at which LSAs are received based on network connections and router resources.

Context

In OSPF, the defined interval at which LSAs are received is 1s. This aims to prevent network connections or frequent route flapping from consuming excessive network bandwidth or device resources.

On a stable network that requires fast route convergence, you can cancel the interval at which LSAs are received by setting the interval to 0s. After the interval is set to 0s, topology or route changes can be immediately advertised on the network through LSAs, which speeds up route convergence.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **lsa-arrival-interval { interval | intelligent-timer max-interval start-interval hold-interval }**

The interval at which LSAs are received is set.

- *interval*: specifies the interval at which LSAs are received, in milliseconds.
- **intelligent-timer**: uses the intelligent timer to set the interval at which OSPF router LSAs and network LSAs are updated.
- *max-interval*: specifies the maximum interval at which OSPF LSAs are received, in milliseconds.
- *start-interval*: specifies the initial interval at which OSPF LSAs are received, in milliseconds.
- *hold-interval*: specifies the hold interval at which OSPF LSAs are received, in milliseconds.

The maximum interval for receiving LSAs is as follows:

1. The initial interval at which LSAs are received is specified by *start-interval*.
2. The interval at which LSAs are received for the nth ($n \geq 2$) time equals *hold-interval* $\times 2^{(n - 1)}$.
3. When the interval specified by *hold-interval* $\times 2^{(n - 1)}$ reaches the maximum interval specified by *max-interval*, OSPF receives LSAs at the maximum interval for three consecutive times. Then, OSPF returns to **Step 3.1** and receives LSAs at the initial interval specified by *start-interval*.

Step 4 (Optional) Run **lsa-arrival-interval suppress-flapping suppress-interval [threshold threshold]**

The suppression period that takes effect when received OSPF LSAs flap is configured.

If no flapping occurs among received OSPF LSAs, the configuration of the **lsa-arrival-interval** command prevents the device from frequently receiving LSAs. If received OSPF LSAs flap, the configuration of the **lsa-arrival-interval suppress-**

flapping command reduces the impact of the flapping on services. If both of them are configured, the device uses the larger value as the flapping suppression time.

Step 5 Run **commit**

The configuration is committed.

----End

Setting the Interval for the SPF Calculation

By setting the interval for the SPF calculation, you can save resources consumed by frequent network changes.

Context

When the OSPF LSDB changes, the shortest path needs to be recalculated. If a network changes frequently, the shortest path is calculated accordingly, which consumes a large number of system resources degrades system performance. By configuring an intelligent timer and a proper interval for the SPF calculation, you can prevent excessive system memory and bandwidth resources from being occupied.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **spf-schedule-interval { interval1 | intelligent-timer max-interval start-interval hold-interval [conservative] | millisecond interval2 }**

The interval for SPF calculation is set.

The interval for SPF calculation based on the intelligent timer is described as follows:

1. The initial interval for SPF calculation is specified by *start-interval*.
2. The interval for SPF calculation for the nth ($n \geq 2$) time equals *hold-interval* $\times 2^{(n - 2)}$.
3. After the interval specified by *hold-interval* $\times 2^{(n - 2)}$ reaches the maximum interval specified by *max-interval*, OSPF keeps using the maximum interval for SPF calculation.
4. If no flapping occurs during the interval from the end of the last SPF calculation to the start of the next SPF calculation, and the interval exceeds the maximum interval specified by *max-interval*, the intelligent timer exits.
5. If no flapping occurs in the previous interval but occurs in the current interval, SPF calculation is delayed for a period of *start-interval*. After the SPF calculation is complete, the current interval will be applied when waiting for the next SPF calculation.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring the Function to Suppress the Advertisement of Interface Addresses

This section describes how to configure the function to suppress the advertisement of interface addresses so that interface addresses can be reused.

Procedure

- Configuring the function to suppress the advertisement of all interface addresses in an OSPF process
 - a. Run **system-view**
The system view is displayed.
 - b. Run **ospf [process-id]**
The OSPF view is displayed.
 - c. Run **suppress-reachability**
The device is configured to suppress the advertisement of all interface addresses in an OSPF process.
 - d. Run **commit**
The configuration is committed.
- Configuring an OSPF interface to suppress the advertisement of interface addresses
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface interface-type interface-number**
The OSPF interface view is displayed.
 - c. Run **ospf suppress-reachability**
An OSPF interface to suppress the advertisement of interface addresses is configured.
 - d. Run **commit**
The configuration is committed.

----End

Configuring the Route Calculation Delay Function to Suppress Frequent LSA Flapping

A route calculation delay can suppress frequent OSPF LSA flapping.

Context

Frequent OSPF LSA flapping on the remote device may lead to route flapping on the local device, affecting services. To address this problem, run the **maxage-lsa route-calculate-delay** command to configure the local device to delay route

calculation in the case of frequent OSPF LSA flapping, which suppresses route flapping locally.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **maxage-lsa route-calculate-delay delay-interval**

The route calculation delay function is configured to suppress frequent OSPF LSA flapping.

Step 4 Run **commit**

The configuration is committed.

----End

Disabling Active/Standby Board Switching Triggered by Abnormal OSPF Aging

By default, active/standby board switching triggered by abnormal OSPF aging is enabled. To disable this function, perform this task.

Context

When the local device's aging timer expires, the local device incorrectly clears all Router LSAs received from the peer device, which causes large-scale route flapping and service interruptions. To resolve this issue, active/standby board switching triggered by abnormal OSPF aging is automatically enabled. Such switching is triggered to restore network connections and service traffic when the following condition is met:

(Number of incorrectly cleared Router LSAs/Total number of Router LSAs) × 100%
≥ 80% (Router LSAs are those sent by the peer device to the local device)

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf maxage-lsa auto-protect disable**

Active/standby board switching triggered by abnormal OSPF aging is disabled.

Step 3 Run **commit**

The configuration is committed.

----End

Disabling OSPF LSA Aging Management

By default, the OSPF LSA aging management function is enabled. To disable this function, perform this task.

Context

LSAs are aged out if their LS age field encounters an exception, and this may cause LSA flapping or incorrect route calculation. For example, the actual LSA aging time of a device is 500 seconds and the device receives an LSA with the aging time of 2500 seconds, the LSA is aged too early because LSAs automatically age in 3600 seconds according to the protocol mechanism. To address this issue, the OSPF LSA aging management function is enabled by default. If the aging time in a received LSA is longer than 1800 seconds, OSPF considers the LSA abnormal and changes the aging time to 1700 seconds. This operation is performed for each abnormal LSA until the aging time values of all LSAs in the area are the same. This ensures that routes can be calculated correctly.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **lsa-age refresh disable**

OSPF LSA aging management is disabled.

Step 4 Run **commit**

The configuration is committed.

----End

Setting a Period During Which OSPF Keeps the Maximum Cost in Local LSAs

If a period during which OSPF keeps the maximum cost in local LSAs is configured and an OSPF interface changes from Down to Up, traffic is switched back only when the period elapses.

Context

When an OSPF interface changes from down to up, the OSPF neighbor relationship is re-established. After IGP route convergence is completed, traffic is switched back. In most cases, IGP routes converge quickly. However, many services that depend on IGP routes may require a delayed switchback. In this case, you can run the **ospf peer hold-max-cost** command to specify a period during which OSPF keeps the maximum cost in local LSAs. After the OSPF neighbor relationship reaches the Full state, the traffic forwarding path remains unchanged during the specified period. After this period expires, the normal cost is restored, and traffic is switched back normally.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The OSPF interface view is displayed.

Step 3 Run **ospf peer hold-max-cost timer timer**

A period during which OSPF keeps the maximum cost in local LSAs is set.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring Secure Synchronization

Secure synchronization prevents traffic loss after a device is restarted.

Context

When the routers in an area just finish synchronizing the LSDBs, the LSDBs of these routers are different from each other. As a result, route flapping occurs. You can configure secure synchronization to solve this problem. This, however, may delay the establishment of the OSPF neighbor relationship.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **safe-sync enable**

Secure synchronization is configured.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration of the OSPF Network Convergence Speed

After configuring OSPF fast network convergence, verify OSPF brief information.

Prerequisites

OSPF fast convergence has been configured.

Procedure

- Run the **display ospf [process-id] brief** command to check brief information about the specified OSPF process.
- Run the **display ospf [process-id] statistics maxage-lsa** command to check information about router LSAs that have reached the aging time.

----End

1.1.4.2.8 Configuring OSPF Delay Advertisement

With OSPF delay advertisement, intra-domain link delay information can be collected and flooded.

Usage Scenario

Traditional path computation algorithms compute the optimal path to a destination address based on costs. However, the path computed in this way may not have the minimum delay. For services that have stringent requirements on delay, path computation can be performed based on the delay rather than on the cost to ensure that service traffic is transmitted along the path with the minimum delay. After OSPF delay advertisement is configured, OSPF collects and floods intra-domain link delay information and reports the information to the controller through BGP-LS. The controller then computes paths on the P2P network based on delay constraints.

Pre-configuration Tasks

Before configuring delay advertisement, configure TWAMP Light to detect delay information. The configuration is as follows:

- Configure the TWAMP Light controller function on the local end.
 - a. Configure the TWAMP Light client function, and create a test session.
 - i. Run **system-view**
The system view is displayed.
 - ii. Run **nqa twamp-light**
The TWAMP Light view is displayed.
 - iii. Run **client**
The TWAMP Light client function is enabled, and the TWAMP Light client view is displayed.
 - iv. Run **test-session session-id sender-ip sender-ip-address reflector-ip reflector-ip-address sender-port sender-port reflector-port reflector-port [dscp dscp-value | padding padding-length | padding-type padding-type | description description] ***
A measurement session is created on the initiator.
 - v. Run **quit**
Return to the TWAMP Light view.
 - b. Configure the TWAMP Light Sender and start the TWAMP Light performance test.

- i. Run **sender**
The TWAMP Light Sender function is enabled, and the TWAMP Light Sender view is displayed.
 - ii. Run **test start-continual test-session session-id [period { 10 | 100 | 1000 | 30000 }] [time-out time-out]**
Continual measurement is started.
 - iii. Run **commit**
The configuration is committed.
- Configure the TWAMP Light Responder on the peer end.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **nqa twamp-light**
The TWAMP Light view is displayed.
 - c. Run **responder**
The TWAMP Light Responder function is enabled, and the TWAMP Light Responder view is displayed.
 - d. Run **test-session session-id local-ip local-ip-address remote-ip remote-ip-address local-port local-port remote-port remote-port interface { interface-type interface-number | interface-name } [anti-loop-on] [description description]**
A measurement session is created on the responder.
 - e. Run **commit**
The configuration is committed.

For details about how to configure TWAMP Light, see TWAMP Light.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

An OSPF process is created, and the OSPF view is displayed.

process-id specifies an OSPF process. If the *process-id* parameter is not specified, the system creates process 1 by default.

Step 3 Run **opaque-capability enable**

The Opaque capability is enabled on the device.

Step 4 Run **metric-delay [average] [variation] advertisement enable**

Delay advertisement is enabled.

Step 5 (Optional) Run **metric-delay normalize interval interval-value [offset offset-value]**

The link delay normalization function is configured for the OSPF process.

For the delay-based path computation algorithm, the delay differences of links are different, and the differences may be small. However, even if the delay differences are small, only one optimal path can be generated according to the existing SPF algorithm, and load balancing cannot be implemented within a certain delay tolerance range. As a result, link resources on the network cannot be fully utilized. To resolve this problem to the greatest extent, normalization processing may be performed on the link delays with a small difference or a difference within an acceptable range so that load balancing can be implemented and link resources on the network can be fully utilized.

Step 6 (Optional) Run **metric-delay suppress timer *time-value* percent-threshold *percent-value* absolute-threshold *absolute-value***

The delay advertisement suppression function is configured.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify the configuration.

- Run the **display ospf interface verbose** command to check information about OSPF-enabled interfaces.

1.1.4.2.9 Configuring OSPF Neighbor Relationship Flapping Suppression

OSPF neighbor relationship flapping suppression works by delaying OSPF neighbor relationship reestablishment or setting the link cost to the maximum value.

Usage Scenario

If an interface carrying OSPF services alternates between Up and Down, OSPF neighbor relationship flapping occurs on the interface. During the flapping, OSPF frequently sends Hello packets to reestablish the neighbor relationship, synchronizes LSDBs, and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, OSPF services, and other OSPF-dependent services, such as LDP and BGP. OSPF neighbor relationship flapping suppression can address this problem by delaying OSPF neighbor relationship reestablishment or preventing service traffic from passing through flapping links.



The following steps are optional, choose them as required.

Pre-configuration Tasks

Before configuring OSPF neighbor relationship flapping suppression, complete the following tasks:

- Configure an IP address for each interface to ensure that neighboring routers are reachable at the network layer.

- **Configure basic OSPF functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 (Optional) Disable OSPF neighbor relationship flapping suppression globally.

1. Run the **ospf [process-id | router-id router-id]*** command to enter the OSPF view.
2. Run the **suppress-flapping peer disable** command to disable OSPF neighbor relationship flapping suppression.

To disable the function globally, perform this step.

Step 3 Run **interface interface-type interface-number**

The interface view is displayed.

Step 4 (Optional) Run **suppress-flapping peer disable**

OSPF neighbor relationship flapping suppression is disabled on the interface.

To disable the function on one of the interfaces, perform this step.

Step 5 Configure an OSPF neighbor relationship flapping suppression mode, which can be Hold-down or Hold-max-cost mode.

- Hold-down mode:

To configure the Hold-down mode and set its duration, run the **ospf suppress-flapping peer hold-down interval** command.

If flooding and topology changes frequently occur during OSPF neighbor relationship establishment, you can configure the Hold-down mode to prevent OSPF neighbor relationship reestablishment within a period of time. This prevents frequent LSDB synchronization and a large number of packets from being exchanged.

- Hold-max-cost mode:

If user service traffic is frequently switched, you can configure OSPF neighbor relationship flapping suppression to work in Hold-max-cost mode. In this mode, the link cost is set to the maximum value (Max-cost) within a period of time, preventing user service traffic from passing through flapping links.

- a. (Optional) Set the maximum cost for OSPF links.

If OSPF neighbor relationship flapping suppression works in Hold-max-cost mode, the device sets the link cost to the maximum value (Max-cost) within a period of time, preventing service traffic from passing through flapping links.

If the value needs to be modified, perform this step.

- i. Run **quit**

Return to the system view.

- ii. Run **ospf [process-id]**

The OSPF view is displayed.

iii. Run **maximum-link-cost cost**

The maximum cost of OSPF is set.

iv. Run **quit**

Return to the system view.

v. Run **interface interface-type interface-number**

The interface view is displayed.

b. (Optional) Run **ospf suppress-flapping peer hold-max-cost disable**

The Hold-max-cost neighbor relationship flapping suppression mode is disabled.

To disable the Hold-max-cost neighbor relationship flapping suppression mode, perform this step.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

Step 6 Run **ospf suppress-flapping peer { detecting-interval *detecting-interval* | threshold *threshold* | resume-interval *resume-interval* }***

Detection parameters are configured for OSPF neighbor relationship flapping suppression.

- Specify an interval for exiting OSPF neighbor relationship flapping suppression.
If the interval between two successive neighbor relationship states (changing from Full to a non-Full state) is longer than the *resume-interval*, the flapping_count is reset.
- If OSPF neighbor relationship flapping suppression works in Hold-max-cost mode, the value of *resume-interval* indicates the duration of this mode.



The value of *resume-interval* must be greater than that of *detecting-interval*.

You can configure detection parameters for OSPF neighbor relationship flapping suppression on specific interfaces according to network conditions. However, using the default values of these parameters is recommended.

Step 7 Run **quit**

The system view is displayed.

Step 8 Run **commit**

The configuration is committed.

Step 9 Run **quit**

The user view is displayed.

Step 10 Run **reset ospf *process-id* suppress-flapping peer [*interface-type interface-number*] [*notify-peer*]**

Interfaces are forced to exit OSPF neighbor relationship flapping suppression.

 NOTE

Interfaces exit flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding OSPF process is reset.
- An OSPF neighbor is reset using the **reset ospf peer** command.
- OSPF neighbor relationship flapping suppression is disabled globally using the **suppress-flapping peer disable** command in the OSPF view.
- The **reset ospf suppress-flapping peer** command is run.

----End

Verifying the Configuration

Run the **display ospf [process-id] interface interfaceType interfaceNum verbose** command to check the status of OSPF neighbor relationship flapping suppression.

1.1.4.2.10 Disabling OSPF Interface Flapping Suppression

OSPF interface flapping suppression is globally enabled by default. If this function is not needed, you can disable it.

Usage Scenario

If OSPF interfaces frequently alternate between up and down, the interfaces will flap, and protocol packets will be frequently exchanged, affecting OSPF services and other services relying on OSPF. Interface flapping suppression can address this issue. This function allows a device to delay interface state changes to up. If this function is not needed, you can disable it.

Pre-configuration Tasks

Before configuring OSPF interface flapping suppression, [configure basic OSPF functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf suppress-flapping interface disable**

OSPF interface flapping suppression is disabled.

Step 3 Run **commit**

The configuration is committed.

----End

Checking the Configuration

Run the **display current-configuration configuration ospf** command to check the status of OSPF interface flapping suppression.

1.1.4.2.11 Configuring Routing Loop Detection for Routes Imported into OSPF

Routing loop detection for routes imported into OSPF enables a device to check whether routes advertised by the device are imported back. If routes advertised by the device are imported back, the device advertises a large link cost for the routes to prevent routing loops.

Usage Scenario

Routing loops may occur when an OSPF process imports routes. If routing loop detection is enabled for routes imported into OSPF on a device and this device detects that it imports a route advertised by itself, it sends this route with a large link cost to other devices. After receiving this route, these devices preferentially select other paths, thereby preventing routing loops.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 (Optional) Run **clear route loop-detect ospf alarm-state**

The routing loop detection alarm state exits, and related alarms are cleared.

NOTE

If the device detects an OSPF routing loop, it reports an alarm. Because the device cannot automatically detect whether the routing loop is eliminated, you need to run this command after the routing loop is eliminated to prevent the device from advertising a large link cost for imported routes and manually clear the OSPF routing loop alarm. If this command is executed when the routing loop has not been eliminated, the alarm is reported again.

Step 3 Run the **route loop-detect ospf enable** command to enable OSPF to detect loops of imported routes. Run the **route loop-detect ospf import-ospf enable** command to enable OSPF to detect loops of imported routes.

NOTE

To reduce the impact of routing loops on services, you are advised to keep routing loop detection enabled for routes imported to OSPF.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.4.2.12 Configuring OSPF Flush Source Tracing

OSPF flush source tracing helps improve troubleshooting efficiency.

Usage Scenario

If network-wide OSPF LSPs are deleted, purge LSPs are flooded, which adversely affects network stability. In this case, source tracing must be implemented to locate the root cause of the fault immediately to minimize the impact. However, OSPF itself does not support source tracing. A conventional solution is isolation

node by node until the faulty node is located, but the solution is complex and time-consuming. To address this problem, enable OSPF flush source tracing.

Pre-configuration Tasks

Before configuring OSPF flush source tracing, complete the following task:

- Configure an IP address for each interface to ensure that neighboring routers are reachable at the network layer.
- [Configure basic OSPF functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 (Optional) Run **ospf flush-source-trace disable**

OSPF flush source tracing is disabled globally.

To disable OSPF flush source tracing globally, perform this step.

Step 3 Run **interface interface-type interface-number**

The interface view is displayed.

Step 4 (Optional) Run **ospf flush-source-trace block**

OSPF flush source tracing is disabled on the interface.

To disable OSPF flush source tracing on an interface, perform this step.

Step 5 Run **quit**

The system view is displayed.

Step 6 Run **ospf flush-source-trace port port-number**

A UDP port number is configured for OSPF flush source tracing packets.

An OSPF flush source tracing port is used to send and receive OSPF flush source tracing packets and is represented by a UDP port number.

Step 7 Run **quit**

The user view is displayed.

Step 8 Run **commit**

The configuration is committed.

Step 9 Run **reset ospf process-id flush-source-trace**

OSPF flush source tracing is reset.

If a large number of OSPF flush source tracing statistics exist on a device, you can run the **reset ospf flush-source-trace** command to reset the statistics and restart OSPF flush source tracing. If the command is run, all dynamic source tracing

statistics are deleted, and the device needs to re-negotiate the source tracing capability with neighboring devices.

----End

Verifying the Configuration

Run the **display ospf [process-id] [area area-id] flush-source-trace analysis-info** command to check information about definitely and possibly faulty nodes identified by OSPF flush source tracing.

1.1.4.2.13 Configuring an OSPF Hostname

Compared with router IDs, OSPF hostnames are easier to memorize. Therefore, using hostnames to identify routers can facilitate network management.

Usage Scenario

To facilitate network management, configure hostnames to identify routers.

Either dynamic or static OSPF hostnames can be configured. In dynamic mode, a hostname can be configured on and advertised by the local device. The mapping between the local device's router ID and hostname can then be queried on a remote router that successfully learns this dynamic hostname.

Pre-configuration Tasks

Before configuring a hostname, complete the following tasks:

- Configure an IP address for each interface to ensure that neighboring routers can use the IP addresses to communicate with each other.
- Configure basic OSPF functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **opaque-capability enable**

The Opaque-LSA function is enabled.

Step 4 Run **hostname host-name**

A dynamic OSPF hostname is configured.

NOTE

If you specify *hostname* in the **hostname** command, the value of *hostname* is advertised as the dynamic hostname. However, if *hostname* is not specified in the command, the hostname specified in the **sysname** command is advertised as the dynamic hostname.

Step 5 Run commit

The configuration is committed.

----End

Verifying the Configuration

Run either of the following commands to check dynamic OSPF hostnames:

- **display ospf [process-id] hostname-table**
- **display ospf [process-id] lsdb [router | network | summary | asbr | ase | nssa | opaque-link | opaque-area] [link-state-id] [originate-router [advertising-router-id] | self-originate | hostname *hostname*] [age { min-value *min-age-value* | max-value *max-age-value* } *]**
- **display ospf [process-id] lsdb [router | network | summary | asbr | ase | nssa | opaque-link | opaque-area] [link-state-id] [originate-router [advertising-router-id] | self-originate] [age { min-value *min-age-value* | max-value *max-age-value* } *] [resolve-hostname]**

1.1.4.2.14 Configuring an OSPF Stub Area

Configuring a non-backbone area as a stub area can reduce routing entries in the area in an AS does not transmit routes learned from other areas in the AS or AS external routes. This reduces bandwidth and storage resource consumption.

Applicable Environment

The number of LSAs can be reduced by partitioning an AS into different areas. To reduce the number of entries in the routing table and the number of LSAs to be transmitted in a non-backbone area, configure the non-backbone area on the border of the AS as a stub area.

Configuring a stub area is optional. A stub area generally resides on the border of an AS. For example, a non-backbone area with only one ABR can be configured as a stub area. In a stub area, the number of entries in the routing table and the amount of routing information to be transmitted greatly decrease.

Note the following points when configuring a stub area:

- The backbone area (Area 0) cannot be configured as a stub area.
- If an area needs to be configured as a stub area, all the routers in this area must be configured with stub attributes using the **stub** command.
- An ASBR cannot exist in a stub area. External routes are not transmitted in the stub area.
- Virtual links cannot exist in the stub area.

Pre-configuration Tasks

Before configuring a stub area, complete the following tasks:

- Configuring IP addresses for interfaces to ensure that neighboring routers are reachable at the network layer

- **Configuring Basic OSPF Functions**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **area area-id**

The OSPF area view is displayed.

Step 4 Run **stub**

The specified area is configured as a stub area.



- All routers in a stub area must have the **stub** command configuration.
- Running the **stub** command or canceling the configuration may cause the stub area to be updated. The **stub** command configuration on a device can be canceled or the configuration can be performed on new devices in the stub area only after the last area update is complete.

Step 5 (Optional) Run **stub no-summary**

The ABR is prevented from sending Type 3 LSAs to the stub area.

Step 6 (Optional) Run **stub default-route-advertise backbone-peer-ignore**

The ABR is prevented from checking the neighbor status when it generates a default Type 3 LSA and advertises it to the stub area.

Step 7 (Optional) Run **default-cost cost**

A cost is set for the default route advertised to the stub area.

To ensure the reachability of AS external routes, the ABR in the stub area generates a default route and advertises it to non-ABR routers in the stub area.

Step 8 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring the stub area, verify the configuration.

- Run the **display ospf [process-id] lsdb** command to check OSPF LSDB information.
- Run the **display ospf [process-id] peer** command to check information about OSPF neighbors.
- Run the **display ospf [process-id] routing** command to check information about the OSPF routing table.

When routers are in a common area, there are AS external routes in the routing table. After the area where the router resides is configured as a stub area, AS external routes no longer exist in the routing table, and the **ASE** field is displayed as 0 in the command output.

1.1.4.2.15 Configuring an NSSA

Configuring a non-backbone area on the border of an AS as an NSSA can reduce entries in the routing table and the amount of routing information to be transmitted. This section describes how to set the cost of the default route to an NSSA and adjust the selection of the default route.

Usage Scenario

An excessive number of entries in a routing table wastes network resources and causes high CPU usage. To reduce entries in a routing table, configure a non-backbone area on the border of an AS as a stub area or an NSSA to reduce the amount of routing information to be transmitted. For details on how to configure an OSPF stub area, see [1.1.4.2.14 Configuring an OSPF Stub Area](#).

An NSSA is a new type of OSPF area. Neither the NSSA nor the stub area transmits routes learned from other areas in the AS where it resides. Different from the stub area, the NSSA allows AS external routes to be imported and forwarded in the entire AS.

If you want to import AS external routes to an area and prevent these routes from consuming resources, configure the area as an NSSA.

Type 7 link state advertisements (LSAs) are used to carry imported AS external routes in the NSSA. Type 7 LSAs are generated by ASBRs of NSSAs and flooded only in the NSSAs where ASBRs reside. The ABR in an NSSA selects Type 7 LSAs from those received, and translates them into Type 5 LSAs in order to advertise external routes to other areas over the OSPF network.

NOTE

- A Type 7 LSA was introduced to support NSSAs and describe imported external routes.
- Default routes can also be represented by Type 7 LSAs to guide traffic to other ASs.

To configure an area as an NSSA, configure NSSA attributes on all the routers in this area.

Pre-configuration Tasks

Before configuring an NSSA, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring routers are reachable at the network layer.
- [Configure basic OSPF functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **area area-id**

The OSPF area view is displayed.

Step 4 Run **nssa [{ default-route-advertise [backbone-peer-ignore] } | no-import-route | no-summary | set-n-bit | suppress-forwarding-address | translator-always | translator-interval interval-value | zero-address-forwarding] ***

The area is configured as an NSSA.

 **NOTE**

- NSSA attributes must be configured on all routers in the NSSA using the **nssa** command.
- Configuring or deleting NSSA attributes may trigger area update and neighbor relationship interruption. You can perform the configuration again or cancel the configuration only after the last area update is complete.

The parameters in the **nssa** command are used in the following scenarios:

- The **default-route-advertise** parameter is used to generate default Type 7 LSAs on the ABR or ASBR and advertise them to the NSSA.
The conditions for generating default Type 7 LSAs and advertising them to the NSSA are as follows:
 - a. Neighbors in Full state and interfaces in Up state must exist in the backbone area.
 - b. The **default-route-advertise** parameter is configured.
 - c. Other default routes (destined for 0.0.0.0) exist in the local routing table.If condition 1 is met, the ABR can generate default Type 7 LSAs and advertise them to the NSSA. If conditions 2 and 3 are met, the ASBR can generate default Type 7 LSAs and advertise them to the NSSA.
- The **backbone-peer-ignore** parameter is used to ignore the check on the neighbor status in the backbone area. That is, the ABR automatically generates default Type 7 LSAs and advertises them to the NSSA as long as an interface that is Up exists in the backbone area, regardless of whether a neighbor in the Full state exists.
- If an ASBR also functions as an ABR, specifying **no-import-route** prevents OSPF from advertising the external routes imported using the **import-route** command to the NSSA.
- To reduce the number of LSAs to be transmitted to the NSSA, specify **no-summary** on an ABR. This prevents the ABR from transmitting Summary LSAs (Type 3 LSAs) to the NSSA.

 **NOTE**

After the **nssa default-route-advertise backbone-peer-ignore no-summary** command is run, the ABR generates both the default Type 7 LSA and default Type 3 LSA as long as the backbone area contains interfaces that are up, regardless of whether neighbor relationships in Full state exist. In this case, the default Type 3 LSA preferentially takes effect.

- After the **set-n-bit** parameter is set, the N-bit is set in the DD packets during the synchronization between the router and neighboring router.
- The **suppress-forwarding-address** parameter sets the forwarding address (FA) of the Type 5 LSAs translated by the NSSA ABR to 0.0.0.0.
- If there are multiple ABRs in an NSSA, the system automatically selects an ABR as the translator according to certain rules (generally, the device with the largest router ID is selected in the NSSA) to translate Type 7 LSAs into Type 5 LSAs. You can configure the **translator-always** parameter on an ABR to specify the ABR as an all-the-time translator. To allow two ABRs to participate in load balancing, configure **translator-always** on the ABRs. You can use this command to pre-configure a fixed translator to prevent LSA flooding caused by translator role changes.
- To ensure service continuity during translator switching, specify the **translator-interval** parameter. The value of *interval-value* must be greater than the flooding interval.
- The **zero-address-forwarding** parameter is used to set the FA of the generated NSSA LSAs to 0.0.0.0 when external routes are imported to the ABR in an NSSA.

Step 5 (Optional) Run **default-cost cost**

The cost of the default route corresponding to Type 3 LSAs that are transmitted by the ABR to the NSSA is set.

To ensure the reachability of AS external routes, the ABR in the NSSA generates a default route and advertises it to other routers in the NSSA. Setting the cost helps control route selection.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run either of the following commands to check LSDB information:

- **display ospf [process-id] lsdb [brief]**
- **display ospf [process-id] lsdb [router | network | summary | asbr | ase | nssa | opaque-link | opaque-area] [link-state-id] [originate-router [advertising-router-id] | self-originate]**

Run either of the following commands to check routing table information:

- **display ospf [process-id] routing [ip-address [mask | mask-length]] [interface interface-type interface-number] [nexthop nexthop-address]**
- **display ospf [process-id] routing router-id [router-id]**

Run the **display ospf [process-id] interface [all | no-peer | interface-type interface-number] [verbose]** command to check OSPF interface information.

By comparing the OSPF routing tables before and after an NSSA is configured, you can reach the following conclusions:

- After an area is configured as an NSSA, the number of entries in the routing table is reduced.
- Different from a stub area, an NSSA can import AS external routes.

1.1.4.2.16 Configuring OSPF Local MT

On a network where multicast and an MPLS TE tunnel are deployed, you can configure OSPF local MT to create a correct multicast routing table and guide multicast packet forwarding.

Usage Scenario

When multicast and an Interior Gateway Protocol (IGP) Shortcut-enabled MPLS TE tunnel are configured on a network, the outbound interface of the route calculated by an IGP may not be a physical interface but a TE tunnel interface. Based on a unicast route to a multicast source address, the router sends a Join message through a TE tunnel interface. In this case, devices spanned by the TE tunnel cannot detect the Join message so that they do not create any multicast forwarding entry.

To resolve this problem, enable OSPF local MT. After local MT is enabled, if the outbound interface of a calculated route is an IGP Shortcut-enabled TE tunnel interface, the route management (RM) module creates an independent Multicast IGP (MIGP) routing table for the multicast protocol, calculates a physical outbound interface for the route, and adds the route to the MIGP routing table. Multicast packets are then forwarded along this route.

Configuring filtering policies for local MT controls the number of entries in the MIGP routing table and speeds up the MIGP routing table lookup.

Pre-configuration Tasks

Before configuring local MT, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring Devices are reachable at the network layer.
- **Configure basic OSPF functions.**
- Configure the IGP Shortcut.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **local-mt enable**

Local MT is enabled.

 NOTE

- Local MT takes effect only in the OSPF process of a public network instance.
- Local MT does not support Forwarding Adjacency (FA).

Step 4 (Optional) Run any of the following commands as required:

- Based on the basic ACL:
 - a. Run **local-mt filter-policy acl { acl-number | acl-name }**
A filtering policy is configured for OSPF local MT.
 - b. Run **quit**
Return to the system view.
 - c. Run **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]**
The ACL view is displayed.
 - d. Run **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] ***
An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID

in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- Based on an IP prefix list:
Run **local-mt filter-policy ip-prefix ip-prefix-name**
A routing policy is configured for OSPF local MT.
- Based on a route-policy:
Run **local-mt filter-policy route-policy route-policy-name**
A route-policy is configured for OSPF local MT.
- Based on a route-filter:
Run **local-mt filter-policy route-filter route-filter-name**
A route-filter is configured for OSPF local MT.

After an MIGP routing table is created, OSPF performs route calculation. If the outbound interface of the calculated route is an IGP Shortcut-enabled TE tunnel interface, the router saves the physical interface on which the tunnel interface is configured as the outbound interface in the MIGP routing table. Multicast packets are then forwarded along this route.

After the routing policy is configured, only the matching routes to the multicast source address are added to the MIGP routing table, which controls the number of entries in the MIGP routing table and speeds up MIGP routing table lookup.

Configure a routing policy before you enable local MT, because if an excessive number of routes to a non-multicast source address exist in an MIGP routing table, the number of entries may exceed the upper limit.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

- Run the **display ospf [process-id] migp-routing [ip-address [mask | mask-length]] [interface interface-type interface-number] [nexthop nexthop-address]** command to check MIGP routing table information.
- Run the **display ospf [process-id] routing** command to check OSPF routing table information.
- Run the **display ospf [process-id] brief** command to check brief OSPF information.

1.1.4.2.17 Configuring an OSPF Sham Link

This section describes how to configure an OSPF sham link so that traffic between sites of the same VPN in the same OSPF area is forwarded through the OSPF intra-area route over the BGP/MPLS IP VPN backbone network.

Usage Scenario

Generally, BGP peers use BGP extended community attributes to carry routing information over the BGP/MPLS IP VPN backbone network. PEs can use the

routing information to exchange inter-area routes between PEs and CEs through OSPF. OSPF sham links are unnumbered P2P links between two PEs over an MPLS VPN backbone network. The source and destination IP addresses of each sham link are IP addresses with a 32-bit mask of loopback interfaces. The loopback interfaces must be bound to a VPN instance, and routes of the two IP addresses are advertised through BGP.

On the BGP/MPLS IP VPN backbone network, if an intra-area OSPF link exists between the network segment where the local CE resides and the network segment where the remote CE resides, the route over this intra-area OSPF link is an intra-area route and has a higher priority than the inter-area route over the BGP/MPLS IP VPN backbone network. In this case, VPN traffic is always forwarded through this intra-area route. To prevent this problem, you can set up an OSPF sham link between the PEs so that the route over the MPLS IP VPN backbone network becomes an OSPF intra-area route and ensure that this route is preferentially selected.

Pre-configuration Tasks

Before configuring an OSPF sham link, complete the following tasks:

- Configure basic BGP/MPLS IP VPN functions and configure OSPF between each PE and its corresponding CE.
- Configure OSPF in the LAN of each CE.

Procedure

Step 1 Configure endpoint IP addresses for a sham link.

Perform the following steps on PEs at both ends of the sham link:

1. Run **system-view**

The system view is displayed.

2. Run **interface loopback *loopback-number***

A loopback interface is created, and its view is displayed.

Each VPN instance must have an endpoint IP address of a sham link, and the IP address is a loopback interface IP address with a 32-bit mask in the VPN address space of a PE. Sham links of the same OSPF process can share the same endpoint IP address. The endpoint IP addresses of sham links of different OSPF processes must be different.

3. Run **ip binding vpn-instance *vpn-instance-name***

A VPN instance is bound to the loopback interface.

4. Run **ip address *ip-address* { *mask* | *mask-length* }**

An IP address is configured for the loopback interface.

 **NOTE**

The configured IP address must have a 32-bit mask (255.255.255.255).

5. Run **commit**

The configuration is committed.

6. Run **quit**

Return to the system view.

Step 2 Advertise routes of the sham link's endpoint IP addresses.

Perform the following steps on PEs at both ends of the sham link:

1. Run **bgp as-number**

The BGP view is displayed.

2. Run **ipv4-family vpn-instance vpn-instance-name**

The BGP-VPN instance IPv4 address family view is displayed.

3. Run **import-route direct**

Import direct routes (routes of the sham link's endpoint IP addresses in this case) to BGP.

Routes of the sham link's endpoint IP addresses are advertised as VPN IPv4 routes by BGP.

 **NOTE**

Ensure that routes of the sham link's endpoint IP addresses are not exchanged by PEs through the VPN OSPF process.

If routes of the sham link's endpoint IP addresses are exchanged by PEs through the VPN OSPF process, each PE has two routes to the other endpoint of the sham link. One of the routes is learned through the VPN OSPF process, and the other is learned through the MP-BGP connection. Because the OSPF route has a higher priority than the BGP route, the OSPF route is selected, causing a sham link establishment failure.

4. Run **commit**

The configuration is committed.

5. Run **quit**

Return to the BGP view.

6. Run **quit**

Return to the system view.

Step 3 Create an OSPF sham link.

Perform the following steps on PEs at both ends of the sham link:

1. Run **ospf [process-id] [router-id router-id | vpn-instance vpn-instance-name] ***

The OSPF multi-instance view is displayed.

2. Run **area area-id**

The OSPF area view is displayed.

3. Run **sham-link source-ip-address destination-ip-address [smart-discover | cost cost | hello hello-interval | dead dead-interval | retransmit retransmit-interval | trans-delay trans-delay-interval | [simple [plain plain-text | cipher cipher-text | cipher-text] | { md5 | hmac-md5 | hmac-sha256 } [key-id { plain plain-text | cipher cipher-text | cipher-text }] | authentication-null | keychain keychain-name]] ***

A sham link is created.

The authentication modes configured at the two ends must be the same. If packet authentication is configured, only the OSPF packets that pass the authentication are accepted; if the authentication fails, the OSPF neighbor relationship cannot be established.

 **NOTE**

To ensure that VPN traffic is forwarded over the MPLS backbone network, ensure that the cost of the sham link is smaller than that of the OSPF route used to forward the traffic over the user network when running the **sham-link** command. In most cases, you need to change the cost of the interfaces on the user network to ensure that the cost of the OSPF route used to forward the traffic over the user network is greater than that of the sham link.

----End

Checking the Configurations

After configuring the OSPF sham link, run the following commands to check the configurations.

- Run the **display ip routing-table vpn-instance *vpn-instance-name*** command on PEs to check the VPN routing table.
- Run the **display ip routing-table** command on CEs to check the routing table.
- Run the **tracert *host*** command on a CE to check the nodes through which data is forwarded to the remote end.
- Run the **display ospf process-id sham-link [area *area-id*]** command on PEs to check whether the sham link is established.
- Run the **display ospf routing** command on CEs to check OSPF routes.

1.1.4.2.18 Configuring BFD for OSPF

After BFD for OSPF is enabled, if a link fails, the router rapidly detects the failure, notifies the OSPF process or interface of the fault, and instructs OSPF to recalculate routes. This speeds up OSPF network convergence.

Usage Scenario

OSPF enables the router to periodically send Hello packets to a neighboring router for fault detection. Detecting a fault takes more than 1s. As voice, video, and other VOD services are widely used. These services are sensitive to packet loss and delays. When traffic is transmitted at gigabit rates, long-time fault detection will cause packet loss. This cannot meet high reliability requirements of the carrier-class network. BFD for OSPF was introduced to resolve this problem. After BFD for OSPF is configured in a specified process or on a specified interface, the link status can be rapidly detected and fault detection can be completed in milliseconds. This speeds up OSPF convergence when the link status changes.

Pre-configuration Tasks

Before configuring BFD for OSPF, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- **Configure basic OSPF functions.**

Configuring BFD Globally

On two devices that need to establish a BFD session with each other, you can configure BFD for all the interfaces in a certain OSPF process.

Context

Perform the following steps on the router that runs OSPF:

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bfd**

BFD is enabled globally.

Step 3 Run **commit**

The configuration is committed.

----End

Configuring BFD for an OSPF Process

Configuring BFD for an OSPF process helps the system rapidly detect link states and speeds up OSPF convergence in the case of a link state change.

Context

After BFD is configured for an OSPF process, when detecting a link failure, BFD rapidly notifies the failure to routers on both ends of the link, triggering rapid OSPF convergence. When the OSPF neighbor relationship goes Down, the BFD session will be dynamically deleted.

Before configuring BFD for OSPF, enable BFD globally.

To configure BFD on all the interfaces in an OSPF process, perform the following operations on the routers at both ends of the link where a BFD session is to be set up:

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **bfd all-interfaces enable**

BFD is enabled for the OSPF process to create BFD sessions.

Step 4 (Optional) Run **bfd all-interfaces { min-rx-interval receive-interval | min-tx-interval transmit-interval | detect-multiplier multiplier-value | frr-binding }** *

BFD session parameters are modified.

You can skip this step. The default interval at which BFD packets are transmitted and the default detection multiplier are recommended.

The parameters are configured based on the network status and network reliability requirements. A short interval at which BFD packets are transmitted can be configured for a link that has a higher requirement for reliability. A long interval at which BFD packets are transmitted can be configured for a link that has a lower requirement for reliability.

NOTE

- Actual interval at which BFD packets are transmitted on the local router = Max { configured interval *transmit-interval* at which BFD packets are transmitted on the local router, configured interval *receive-interval* at which BFD packets are received on the peer router }
- Actual interval at which BFD packets are received on the local router = Max { configured interval *transmit-interval* at which BFD packets are transmitted on the peer router, configured interval *receive-interval* at which BFD packets are received on the local router }
- Actual time for detecting BFD packets = Actual interval at which BFD packets are received on the local router x Configured detection multiplier *multiplier-value* on the peer router

For example:

- On the local router, the configured interval at which BFD packets are transmitted is 200 ms; the configured interval at which BFD packets are received is 300 ms; the detection multiplier is 4.
- On the peer router, the configured interval at which BFD packets are transmitted is 100 ms; the interval at which BFD packets are received is 600 ms; the detection multiplier is 5.

Then:

- On the local router, the actual interval at which BFD packets are transmitted is 600 ms calculated by using the formula max {200 ms, 600 ms}; the interval at which BFD packets are received is 300 ms calculated by using the formula max {100 ms, 300 ms}; the detection period is 1500 ms calculated by multiplying 300 ms by 5.
- On the peer router, the actual interval at which BFD packets are transmitted is 300 ms calculated by using the formula max {100 ms, 300 ms}, the actual interval at which BFD packets are received is 600 ms calculated by using the formula max {200 ms, 600 ms}, and the detection period is 2400 ms calculated by multiplying 600 ms by 4.

Step 5 (Optional) Prevent an interface from dynamically creating a BFD session.

After BFD for OSPF is configured, all interfaces on which neighbor relationships are Full in the OSPF process will create BFD sessions. To prevent specific interfaces from being enabled with BFD, disable these interfaces from dynamically creating BFD sessions.

1. Run **quit**

Return to the system view.

2. Run **interface *interface-type interface-number***

The interface view is displayed.

3. Run **ospf bfd block**

An interface is prevented from dynamically creating a BFD session.

4. Run **quit**

Return to the system view.

5. Run **ospf [process-id]**

The OSPF view is displayed.

Step 6 (Optional) Run **bfd all-interfaces incr-cost { cost | max-reachable } [wtr <wtr-value>]**

The OSPF process is enabled to adjust the cost based on the BFD session status.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

Step 7 Run **commit**

The configuration is committed.

----End

(Optional) Configuring BFD for a Specified Interface

Configuring BFD for OSPF on a specified interface helps speed up OSPF convergence in the case of an interface failure.

Context

After BFD for OSPF is configured on a specified interface and the interface becomes faulty, the router rapidly detects the fault and instructs OSPF to recalculate routes. This speeds up OSPF convergence. When the OSPF neighbor relationship goes Down, the BFD session between OSPF neighbors is dynamically deleted.

Before configuring BFD for OSPF, enable BFD globally.

Perform the following steps on the router where a BFD session is to be established on a specified interface:

 **NOTE**

The interface can be a physical interface or a GRE tunnel interface. If BFD is enabled on a GRE tunnel interface, millisecond-level fault detection can be implemented for the GRE tunnel.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bfd**

BFD is globally configured.

Step 3 Run **quit**

Return to the system view.

Step 4 Run **interface interface-type interface-number**

The interface view is displayed.

Step 5 Run **ospf bfd enable**

BFD for OSPF is configured, and the default parameter values are used to establish a BFD session.

If BFD is enabled globally and the neighbor relationships on the interface are in the Full state, OSPF creates a BFD session with default parameter values for the interface.

The **ospf bfd enable [per-link one-arm-echo]** command can be run only in the VLANIF interface view.

If BFD for OSPF is configured for an Eth-Trunk with multiple physical interfaces added in a VLAN and **per-link one-arm-echo** is not specified, the BFD session may go down even if only one of the physical interfaces goes down. As a result, the OSPF neighbor relationship also goes down. If **per-link one-arm-echo** is specified in this case, the BFD session goes down only if all the physical interfaces are down, which ensures that the OSPF neighbor relationship can be established normally.

 **NOTE**

The configuration of BFD for OSPF on an interface takes precedence over that in the OSPF process.

Step 6 (Optional) Run **ospf bfd { min-rx-interval receive-interval | min-tx-interval transmit-interval | detect-multiplier multiplier-value | frr-binding } ***

BFD session parameters are modified.

The default interval at which BFD packets are transmitted and the default detection multiplier are recommended. As such, this step can be skipped.

The parameters need to be configured based on network conditions and requirements on network reliability. A short transmission interval for BFD packets can be set for a link that requires higher reliability, and a long transmission interval can be configured for a link that requires lower reliability.

 NOTE

- Actual interval at which BFD packets are transmitted on the local device = Max { *transmit-interval* (interval at which BFD packets are transmitted) set on the local device, *receive-interval* (interval at which BFD packets are received) set on the peer device }
- Actual interval at which BFD packets are received on the local device = Max { *transmit-interval* (interval at which BFD packets are transmitted) set on the peer device, *receive-interval* (interval at which BFD packets are received) set on the local device }
- Actual period for BFD detection on the local device = Actual interval at which BFD packets are received on the local device x Detection multiplier specified by *multiplier-value* on the peer device

For example:

- On the local device, the interval at which BFD packets are transmitted is set to 200 ms, the interval at which BFD packets are received is set to 300 ms, and the detection multiplier is set to 4.
- On the peer device, the interval at which BFD packets are transmitted is set to 100 ms, the interval at which BFD packets are received is set to 600 ms, and the detection multiplier is set to 5.

Then:

- On the local device, the actual interval at which BFD packets are transmitted is 600 ms (calculated through Max { 200 ms, 600 ms }); the actual interval at which BFD packets are received is 300 ms (calculated through Max { 100 ms, 300 ms }); the actual detection period is 1500 ms (calculated through 300 ms x 5).
- On the peer device, the actual interval at which BFD packets are transmitted is 300 ms (calculated through Max { 100 ms, 300 ms }); the actual interval at which BFD packets are received is 600 ms (calculated through Max { 200 ms, 600 ms }); the actual detection period is 2400 ms (calculated through 600 ms x 4).

Step 7 (Optional) Run **ospf bfd incr-cost { cost | max-reachable } [wtr <wtr-value>]**

The OSPF interface is enabled to adjust the cost based on the BFD session status.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

Step 8 Run **commit**

The configuration is committed.

----End

Verifying the Configuration of BFD for OSPF

After BFD for OSPF is configured, you can check information about the BFD session.

Prerequisites

BFD for OSPF has been configured.

Procedure

- Run either of the following commands to check the BFD session:
 - **display ospf [process-id] bfd session interface-type interface-number [router-id]**
 - **display ospf [process-id] bfd session { router-id | all }**
- End

1.1.4.2.19 Configuring OSPF IP FRR

If a link fails, an OSPF IP FRR-capable device can fast switch traffic to a backup link, which protects traffic and improves OSPF network reliability.

Usage Scenario

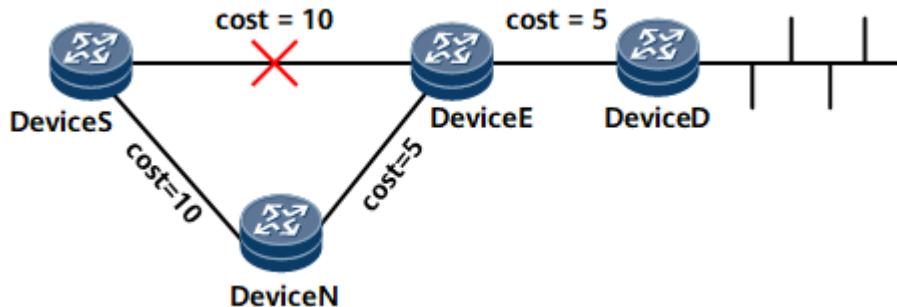
As networks develop, services such as Voice over IP (VoIP) and on-line video services require high-quality real-time transmission.

Nevertheless, if an OSPF fault occurs, traffic can be switched to a new link only after the fault detection that lasts several milliseconds, fault notification to the routing control plane that lasts several milliseconds, new topology information generation and flooding that last tens of milliseconds, Shortest Path First (SPF) calculation that lasts tens of milliseconds, and new route notification and adding that last hundreds of milliseconds. As a result, it takes much more than 50 ms, the maximum convergence time tolerable for VoIP and on-line video services, which cannot meet the requirement for real-time services on the network.

With OSPF IP FRR that calculates a backup link in advance, devices can fast switch traffic to the backup link without interrupting traffic if the primary link fails, which protects traffic and improves OSPF network reliability.

As shown in [Figure 1-122](#), traffic flows from Device S to Device D. The preceding inequality is met. Device S switches the traffic to the backup link if the primary link fails.

Figure 1-122 OSPF IP FRR link protection



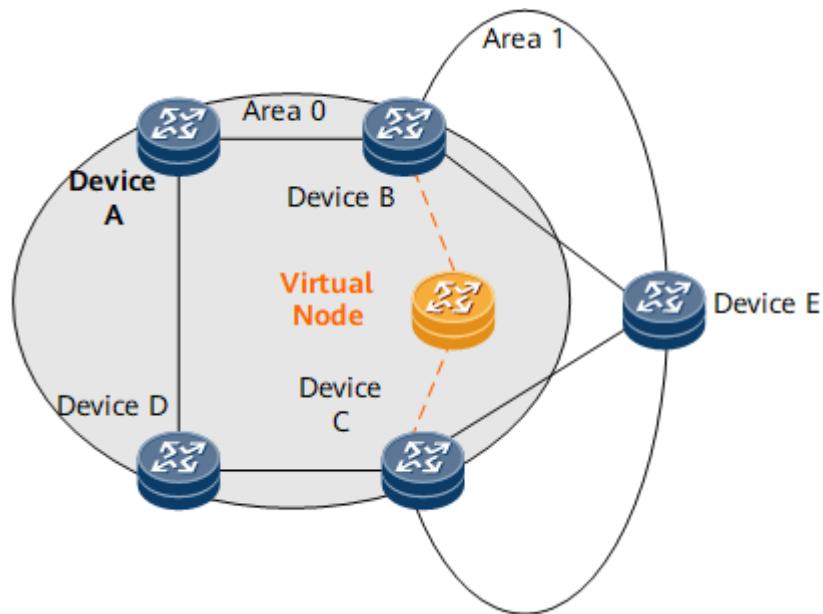
OSPF IP FRR is applicable to services that are sensitive to packet delay and packet loss.

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports OSPF IP FRR that uses LFA or Remote LFA as the algorithm.

LFA Auto FRR cannot be used to calculate alternate links on large-scale networks, especially on ring networks. To address this problem, enable Remote LFA Auto FRR. Remote LFA takes effect only when LFA has been enabled.

Both OSPF LFA FRR and OSPF remote LFA FRR use the SPF algorithm to calculate the shortest path to the destination node, with each neighbor that provides a backup link as the root node. The backup next hop is node-based, which applies to single-node routing scenarios. As networks are increasingly diversified, two ABRs or ASBRs are deployed to improve network reliability. In this case, OSPF FRR in a multi-node routing scenario is needed. In [Figure 1-123](#), Device B and Device C function as ABRs to forward area 0 and area 1 routes. Device E advertises an intra-area route. Upon receipt of the route, Device B and Device C translate it to a Type 3 LSA and flood the LSA to area 0. After OSPF FRR is enabled on Device A, Device A considers Device B and Device C as its neighbors. Without a fixed neighbor as the root node, Device A fails to calculate FRR backup next hop. To solve this problem, a virtual node is constructed between Device B and Device C. The virtual node converts the same route advertised by multiple nodes into a route with a single route source, and then calculates the backup next hop of the virtual node based on the LFA or Remote LFA algorithm. The same route advertised by multiple nodes inherits the backup next hop from the created virtual node.

Figure 1-123 OSPF FRR in the scenario where multiple nodes advertise the same route



After OSPF IP FRR is configured, the lower layer needs to fast respond to a link change so that traffic can be fast switched to the backup link. After FRR and BFD are bound, link failures can be detected rapidly so that traffic is rapidly switched to the backup link if the primary link fails.

Pre-configuration Tasks

Before configuring OSPF IP FRR, complete the following tasks:

- Configure a link layer protocol.

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic OSPF functions.**
- Before you configure remote LFA FRR, configure LDP LSPs to perform recursion hop by hop between the source node and PQ node. That is, configure a local LDP session between each pair of directly connected nodes along the link from the source node to PQ node.

Enabling OSPF IP FRR

With OSPF IP FRR and loop-free backup links, a device can switch traffic to a backup link immediately if the primary link fails.

Context

Perform the following steps on the router:

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id | router-id router-id | vpn-instance vpn-instance-name]** *

An OSPF process is started, and the OSPF view is displayed.

Step 3 Run **frr**

The OSPF FRR view is displayed.

Step 4 Run **loop-free-alternate**

OSPF IP FRR is enabled to generate a loop-free backup link.

Step 5 (Optional) Run **frr-policy route { route-policy route-policy-name | route-filter route-filter-name }**

An OSPF IP FRR filtering policy is configured.

After the OSPF IP FRR filtering policy is configured, only the OSPF backup routes that match the filtering conditions of the policy can be added to the forwarding table.

Step 6 If you want to configure remote LFA OSPF IP FRR, perform the following steps:

- Run **remote-lfa tunnel ldp [maximum-reachable-cost cost-value]**

Remote LFA OSPF IP FRR is enabled.

- (Optional) Run **remote-lfa available-tunnel-destination ip-prefix ip-prefix-name**

A filtering policy is configured to filter PQ nodes.

After a filtering policy is configured, only the nodes that meet the filtering conditions become PQ nodes, which facilitates network optimization.

- Run **quit**

Return to the OSPF view.

4. (Optional) Run **avoid-microloop frr-protected**

The OSPF microloop avoidance is enabled.

5. (Optional) Run **avoid-microloop frr-protected rib-update-delay rib-update-delay**

The delay after which OSPF delivers routes is configured.

If OSPF remote LFA FRR is enabled and the primary link fails, traffic is switched to the backup link. If route convergence occurs again, traffic is switched from the backup link to a new primary link. During the switchover, microloop may occur. To prevent this problem, OSPF anti-microloop is enabled and delays the switching. To configure the delay, run the **avoid-microloop frr-protected rib-update-delay** command. After the command is run, OSPF does not switch traffic to the backup link until the delay elapses.

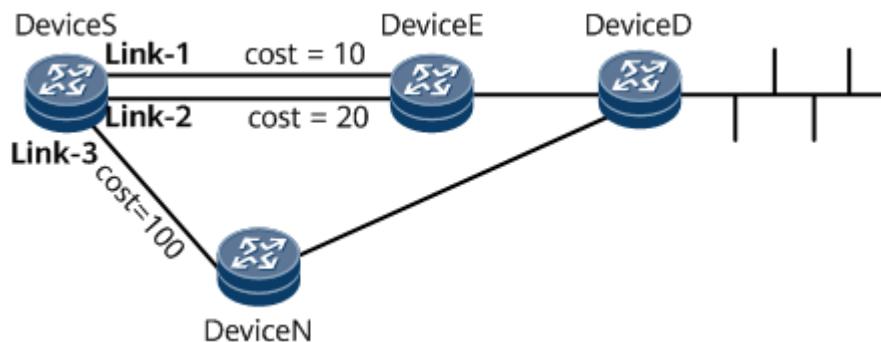
6. Run **frr**

The OSPF FRR view is displayed.

Step 7 (Optional) Run the **tiebreaker { node-protecting | lowest-cost | ldp-sync hold-max-cost | srlg-disjoint } preference** command to configure the solution of selecting a backup path for OSPF IP FRR.

By default, the solution of selecting a backup path for OSPF IP FRR is node-protection path first. In actual networking scenarios, the solution may need to be changed to smallest-cost path first due to considerations such as interface forwarding capacity and link cost. In [Figure 1-124](#), the primary path is Link-1 (DeviceS -> DeviceE -> DeviceD), and Link-2 and Link-3 (DeviceS -> DeviceN -> DeviceD) are backup path candidates. By default, Link-3 is selected as the backup path. To change the solution of selecting a backup path for OSPF IP FRR to smallest-cost path first, run the **tiebreaker** command. After the command is run, Link-2 is selected as the backup path.

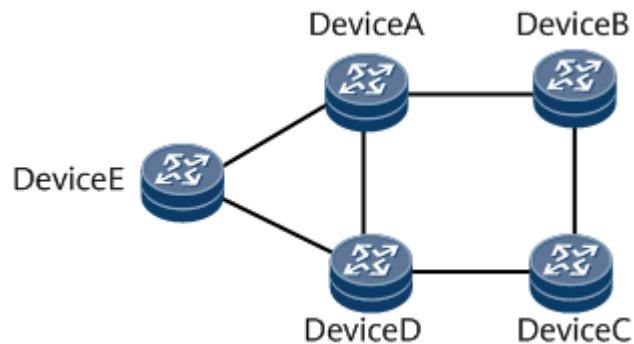
Figure 1-124 Solution of selecting a backup path for OSPF IP FRR



[Figure 1-125](#) shows an inter-board scenario, where Link-1 (Device A -> Device D) is the primary path, and Link-2 (Device A -> Device E -> Device D) is the backup path. After the primary path Link-1 fails, the primary path is switched to Link-2, and the backup path is Link-3 (Device A->Device B->Device C->Device D). If Link-1 goes Up again but the LDP session has not gone Up, OSPF enters the Hold-max-cost state. Consequently, the primary path is still Link-2, and the backup path is still Link-3. If the LDP session goes Up but **ldp-sync hold-max-cost** is not configured, OSPF exits from the Hold-max-cost state when the timer used to

delay sending an LDP session Up message expires. In this case, OSPF switches the primary path back to Link-1. Because the upstream and downstream entries reside on different boards and the downstream entry has not been updated when downstream traffic arrives, packet loss occurs. To resolve the problem, configure **ldp-sync hold-max-cost** so that OSPF preferentially selects the path with the maximum cost set by LDP-IGP synchronization when OSPF is in the Hold-max-cost state. The backup path is switched to Link-1, and backup forwarding entries are delivered in advance. When the timer used to delay sending an LDP session Up message expires, OSPF exits from the Hold-max-cost state and switches the primary path to Link-1. Because the downstream backup entry is available, no packet loss occurs.

Figure 1-125 Maximum-cost (set by LDP-IGP synchronization) path first solution



Step 8 Run commit

The configuration is committed.

----End

(Optional) Binding IP FRR and BFD

Binding IP FRR and BFD enables the lower layer to fast respond to a link change so that traffic can be rapidly switched to the backup link if the primary link fails.

Context

After the parameter **frr-binding** is set to bind the BFD status to the link status of an interface, link failures can be detected rapidly. This ensures that traffic is rapidly switched to the backup link if the primary link fails.

Perform the following steps on the router where IP FRR and BFD need to be bound:

Procedure

- Bind IP FRR and BFD in an OSPF process.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **ospf**
An OSPF process is started, and the OSPF view is displayed.

- c. Run **bfd all-interfaces frr-binding**
IP FRR and BFD are bound in the OSPF process.
 - d. Run **commit**
The configuration is committed.
- Bind IP FRR and BFD on a specified OSPF interface.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface interface-type interface-number**
The interface view is displayed.
 - c. Run **ospf bfd frr-binding**
IP FRR and BFD are bound on the interface.

 **NOTE**

The BFD configuration on an interface takes precedence over that in an OSPF process.

- d. Run **commit**
The configuration is committed.

----End

(Optional) Disabling OSPF IP FRR on an Interface

If an interface runs key services, ensure that the backup path does not pass through this interface so that services will not be affected after FRR calculation.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The view of an OSPF interface on which FRR is enabled is displayed.

Step 3 Run **ospf frr block**

FRR is blocked on the OSPF interface.

Step 4 (Optional) Run **ospf remote-lfa disable**

Interfaces of the specified level are disabled from being calculated as the Remote LFA next hop.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration of OSPF IP FRR

After configuring OSPF IP FRR, you can view the information about the backup next hop.

Prerequisites

OSPF IP FRR has been configured.

Procedure

- Run the **display ospf [process-id] routing** command to check the information about the primary and backup links after configuring OSPF IP FRR.

----End

1.1.4.2.20 Configuring OSPF GR Helper

To avoid traffic interruption and route flapping caused by the active/standby switchover, you can enable OSPF GR.

Usage Scenario

Graceful Restart (GR) is a technology that ensures normal data forwarding without affecting key services when a routing protocol restarts. GR is one of high availability (HA) technologies. HA technologies comprise a set of comprehensive techniques, such as fault-tolerant redundancy, link protection, faulty node recovery, and traffic engineering. As a fault-tolerant redundancy technology, GR is widely used to ensure non-stop forwarding of key data during the active/standby switchover and system upgrade.



GR involves two roles: the GR restarter and GR helper. The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports the GR helper only.

Pre-configuration Tasks

Before configuring OSPF GR, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic OSPF functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **opaque-capability enable**

The Opaque-LSA capability is enabled.

OSPF supports OSPF GR through Type 9 LSAs. Therefore, before configuring GR, run the **opaque-capability enable** command to enable the Opaque-LSA capability.

Step 4 Configure GR session parameters on the helper.

1. Run **graceful-restart helper-role ignore-external-lsa**

The Helper is configured to ignore AS-external LSAs.

2. Run **graceful-restart helper-role planned-only**

The Helper is configured to support only planned GR.

3. Choose either of the following configuration as required:

- Based on a basic ACL:

- i. Run **graceful-restart [helper-role { { acl-number *acl-number* | acl-name *acl-name* } * } | never]**

GR session parameters on the helper are configured.

- ii. Run **quit**

The system view is displayed.

- iii. Run **acl { name *basic-acl-name* { basic | [basic] number *basic-acl-number* } | [number] *basic-acl-number* } [match-order { config | auto }]**

The ACL view is displayed.

- iv. Run **rule [rule-id] [name *rule-name*] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range *time-name* | vpn-instance *vpn-instance-name*] ***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

- Configure an IP prefix list.

- Run **graceful-restart [helper-role { { ip-prefix *ip-prefix-name* * } | never]**

GR session parameters on the helper are configured.

Step 5 Configure GR session parameters on the helper.

 **NOTE**

IETF mode and non-IETF mode are mutually exclusive.

- Configure GR session parameters on the helper in IETF mode.

- a. Run **graceful-restart helper-role ignore-external-lsa**

The helper does not check AS external LSAs.

- b. Run **graceful-restart helper-role planned-only**

The helper is configured to support only planned GR.

- c. Choose either of the following configuration as required:
- Based on a basic ACL:
 - 1) Run **graceful-restart [helper-role { { acl-number *acl-number* | acl-name *acl-name* }* } | never }**
GR session parameters on the helper are configured.
 - 2) Run **quit**
The system view is displayed.
 - 3) Run **acl { name *basic-acl-name* { basic | [basic] number *basic-acl-number* } | [number] *basic-acl-number* } [match-order { config | auto }]**
The ACL view is displayed.
 - 4) Run **rule [rule-id] [name *rule-name*] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first }] | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range *time-name* | vpn-instance *vpn-instance-name*]***
The rule for the ACL is configured.
When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.
 - Configure an IP prefix list.
Run **graceful-restart [helper-role { { ip-prefix *ip-prefix-name** } | never }]**
GR session parameters on the helper are configured.
 - Enable the non-IETF mode.
Run **graceful-restart non-ietf**
GR Helper in non-IETF mode is enabled.

Step 6 Run commit

The configuration is committed.

----End

Verifying the Configuration

Run the **display ospf graceful-restart** command, and you can view the configurations about OSPF GR.

1.1.4.2.21 Improving OSPF Network Security

On a network demanding high security, you can configure OSPF authentication and GTSM to improve OSPF network security.

Usage Scenario

With the increase in attacks on TCP/IP networks and the defects in the TCP/IP protocol suite, network attacks have increasing impacts on the network security.

Attacks on network devices may lead to network crash. Configuring OSPF authentication and GTSM can improve OSPF network security.

OSPF authentication encrypts OSPF packets by adding the authentication field to packets to ensure network security. When receiving an OSPF packet from a remote device, the local device discards the packet if the authentication password carried in the packet is different from the local one, protecting the local device against potential attacks.

In terms of the packet type, the authentication is classified as follows:

- Area authentication: configured in the OSPF area view and applies to packets on all interfaces in the OSPF area.
- Interface authentication: configured in the interface view and applies to all packets on the interface.

Pre-configuration Tasks

Before improving OSPF network security, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- **Configure basic OSPF functions.**
- Configure basic Keychain functions.

Configuring Area Authentication

OSPF supports packet authentication. With the authentication, only the packets that are authenticated are accepted; if packets fail to be authenticated, a neighbor relationship cannot be established. If area authentication is used, the authentication mode and password configurations on all the interfaces in the area must be identical.

Context



By default, no authentication mode is configured for an OSPF area. You are advised to configure an authentication mode to ensure system security.

For security purposes, you are advised not to use weak security algorithms in OSPF. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id]**

The OSPF view is displayed.

Step 3 Run **area area-id**

The OSPF area view is displayed.

Step 4 Run any of the following commands to configure the authentication mode for the OSPF area as required:

- Run **authentication-mode simple [plain plain-text | [cipher] cipher-text]**
Simple authentication is configured for the OSPF area.
 - **plain** indicates the plaintext mode.
 - **cipher** indicates the ciphertext mode. For MD5, HMAC-MD5, or HMAC-SHA256 authentication, **cipher** is used by default.
- When configuring an authentication password, you are advised to use the ciphertext mode. If you select the plaintext mode, the password is saved in configuration files in plaintext, which poses a high security risk. To ensure device security, change the password periodically.
- Run **authentication-mode { md5 | hmac-md5 | hmac-sha256 } [key-id { plain plain-text | [cipher] cipher-text }]**
Cipher-text authentication is configured for the OSPF area.
 - **md5** indicates the MD5 ciphertext authentication mode.
 - **hmac-md5** indicates the HMAC-MD5 ciphertext authentication mode.
 - **hmac-sha256** indicates the HMAC-SHA256 ciphertext authentication mode.

 **NOTE**

For the sake of security, using the HMAC-SHA256 algorithm rather than the MD5 or HMAC-MD5 algorithm is recommended.

- Run **authentication-mode keychain Keychain-Name**
The Keychain authentication is configured for the OSPF area.

 **NOTE**

Before using the Keychain authentication, you must run the **keychain** command to create a keychain. Then, run the **key-id**, **key-string**, and **algorithm** commands to configure a key ID, a password, and an authentication algorithm for this keychain. Otherwise, the OSPF authentication will fail.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring Interface Authentication

Interface authentication takes effect between neighboring devices' interfaces on which an authentication mode and password are configured. Interface authentication takes precedence over area authentication. Interfaces on the same network segment must have the same authentication mode and password. Interfaces on different network segments can have different authentication modes and passwords.

Context

NOTE

By default, no authentication mode is configured for an OSPF interface. You are advised to configure an authentication mode to ensure system security.

For security purposes, you are advised not to use weak security algorithms in OSPF. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The OSPF interface view is displayed.

Step 3 Run any of the following commands to configure interface authentication as required:

- Run **ospf authentication-mode simple [plain plain-text | [cipher] cipher-text]**

Simple authentication is configured for the OSPF interface.

- **simple** indicates simple authentication.
- **plain** indicates the plaintext mode. In simple authentication mode, if this parameter is not specified, the **cipher** type is used by default.
- **cipher** indicates the ciphertext mode. For MD5, HMAC-MD5, or HMAC-SHA256 authentication, **cipher** is used by default.

When configuring an authentication password, you are advised to use the ciphertext mode. If you select the plaintext mode, the password is saved in configuration files in plaintext, which poses a high security risk. To ensure device security, change the password periodically.

- Run **ospf authentication-mode { md5 | hmac-md5 | hmac-sha256 } [key-id { plain plain-text | [cipher] cipher-text }]**

Ciphertext authentication is configured for the OSPF interface.

- **md5** indicates the MD5 ciphertext authentication mode.
- **hmac-md5** indicates the HMAC-MD5 ciphertext authentication mode.
- **hmac-sha256** indicates the HMAC-SHA256 ciphertext authentication mode.

NOTE

For the sake of security, using the HMAC-SHA256 algorithm rather than the MD5 or HMAC-MD5 algorithm is recommended.

- Run **ospf authentication-mode keychain keychain-name**

The Keychain authentication is configured for the OSPF interface.

 NOTE

Before using keychain authentication, run the **keychain** command to create a keychain. Then, run the **key-id**, **key-string**, and **algorithm** commands to configure a key ID, a password, and an authentication algorithm, respectively, for this keychain. If a keychain or its parameters are not configured, the OSPF authentication will fail.

- Run **ospf authentication-mode null**

The OSPF interface does not perform authentication.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration of OSPF Network Security

After configuring OSPF functions to improve OSPF network security, check the configuration.

Prerequisites

OSPF functions have been configured to improve OSPF network security.

Procedure

- Run the **display this** command to view the configurations of the system in the current view.

----End

1.1.4.2.22 Configuring the Network Management Function of OSPF

OSPF supports the network management function. You can bind the OSPF MIB to an OSPF process and configure trap and log functions.

Usage Scenario

Through the Simple Network Management Protocol (SNMP), the OSPF Management Information Base (MIB) manages information exchanged between the NMS and agents.

Pre-configuration Tasks

Before configuring the network management function of OSPF, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPF functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf mib-binding process-id**

The OSPF process is bound to the MIB.

Step 3 Run **commit**

The configuration is committed.

----End

Checking the Configurations

Run the following commands to check the previous configuration.

Run the **display ospf [process-id] brief** command to view information about the binding of OSPF MIBs and OSPF processes.

1.1.4.2.23 Configuring Whitelist Session-CAR for OSPF

You can configure whitelist session-CAR for OSPF to isolate bandwidth resources by session for OSPF packets. This configuration prevents bandwidth preemption among OSPF sessions in the case of a traffic burst.

Context

When OSPF packets suffer a traffic burst, bandwidth may be preempted among OSPF sessions. To resolve this problem, you can configure whitelist session-CAR for OSPF to isolate bandwidth resources by session. If the default parameters of whitelist session-CAR for OSPF do not meet service requirements, you can adjust them as required.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **whitelist session-car ospf { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } ***

Parameters of whitelist session-CAR for OSPF are configured.

In normal cases, you are advised to use the default values of these parameters.

Step 3 (Optional) Run **whitelist session-car ospf disable**

Whitelist session-CAR for OSPF is disabled.

By default, whitelist session-CAR for OSPF is enabled. In normal cases, you are advised to keep this function enabled. Disable it if it becomes abnormal or adversely affects other services.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring whitelist session-CAR for OSPF, verify the configuration.

- Run the **display cpu-defend whitelist session-car ospf statistics slot slot-id** command to check statistics about whitelist session-CAR for OSPF on a specified interface board.

1.1.4.2.24 Configuring Micro-Isolation CAR for OSPF

By default, micro-isolation CAR is enabled for OSPF, implementing micro-isolation protection for OSPF packets based on interfaces and destination IP addresses to protect session establishment.

Context

By default, micro-isolation CAR is enabled for OSPF, implementing micro-isolation protection for OSPF packets based on interfaces and destination IP addresses to protect session establishment. When OSPF packets encounter a traffic burst, a large number of OSPF packets may preempt interface bandwidth resources. In the case of an attack, a large number of invalid packets sent to other IP addresses may preempt the bandwidth. Therefore, you are advised not to disable this function.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **micro-isolation protocol-car ospf { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } ***

Micro-isolation CAR parameters are configured for OSPF.

In normal cases, you are advised to use the default values of these parameters. *pir-value* must be greater than or equal to *cir-value*, and *pbs-value* must be greater than or equal to *cbs-value*.

Step 3 (Optional) Run **micro-isolation protocol-car ospf disable**

Micro-isolation CAR is disabled for OSPF.

By default, micro-isolation CAR for OSPF is enabled. To disable this function, run the **micro-isolation protocol-car ospf disable** command. If this function is disabled, micro-isolation protection is no longer implemented for OSPF packets based on interfaces and destination IP addresses. In normal cases, you are advised to keep micro-isolation CAR enabled for OSPF.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.4.2.25 Configuring the Threshold Alarm Function for the Number of OSPF Neighbors

If the alarm function is configured for the number of OSPF neighbors and the number of OSPF neighbors exceeds the threshold, an alarm is generated.

Context

When the number of established OSPF neighbor relationships exceeds the specifications of the device, establishing more OSPF neighbor relationships may cause the neighbor status to be unstable. To prevent this problem, you can enable the alarm function for the number of OSPF neighbor relationships so that an alarm is generated when the number of OSPF neighbor relationships exceeds the upper alarm threshold.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf peer alarm-threshold *threshold-value upper-limit upper-value lower-limit lower-value***

The threshold alarm function is configured for the number of OSPF neighbors.

Step 3 Run **commit**

The configuration is committed.

----End

1.1.4.2.26 Maintaining OSPF

Maintaining OSPF involves resetting OSPF and clearing OSPF statistics.

Clearing OSPF Statistics

This section describes how to clear OSPF statistics, including statistics about OSPF counters and statistics about session-CAR.

Context

NOTICE

OSPF information cannot be restored after you clear it. Exercise caution when clearing it.

To clear OSPF information, run the following reset commands in the user view.

Procedure

- Run the **reset ospf [*process-id*] counters [**neighbor** [*interface-type interface-number*] [*router-id*]]** command to reset OSPF counters.

- **counters** indicates OSPF counters.
- **neighbor** indicates neighbor information on the specified interface.
- Run the **reset ospf [process-id] counters maxage-lsa** command to delete the statistics about router LSAs that have reached the aging time.
- Run the **reset ospf [process-id] frr** command to perform OSPF IP FRR calculation again.
- Run the **reset ospf [process-id] peer [interface-type interface-number] router-id** command to restart an OSPF neighbor.
- Run the **reset cpu-defend whitelist session-car ospf statistics slot slot-id** command to clear statistics about whitelist session-CAR for OSPF on a specified interface board.

----End

Resetting OSPF

You can reset OSPF by restarting OSPF.

Context

NOTICE

Resetting an OSPF connection interrupts the OSPF adjacency between devices. Exercise caution when running the command.

To reset OSPF connections, perform the following operation:

Procedure

- Run the **reset ospf [process-id] process** command to restart the OSPF process.

After the **reset ospf process** command is used to restart OSPF, the following situations may occur:
 - If the router ID is changed, the new router ID takes affect after the **reset ospf process** command is run.
 - The DR and BDR are re-selected after the **reset ospf process** command is run.
- Run the **reset ospf [process-id] spf** to recalculate OSPF routes.
- Run the **reset ospf [process-id] redistribution** to reset OSPF route redistribution.

----End

Disabling OSPF Memory Overload Control

The OSPF memory overload control function is enabled by default. You can disable this function if necessary.

Context

By default, OSPF memory overload control is enabled. When the system memory is overloaded, each module needs to take necessary measures to control the memory usage. Upon receiving a memory overload notification from the system, the OSPF module no longer imports new routes and starts to control neighbor relationship establishment based on the memory overload condition to enhance OSPF resilience. In this case, new neighbor relationships cannot be established. For existing neighbor relationships, if they are in the Full state, they will be retained; if they are in a non-Full state, they are not reestablished until the memory recovers from overload.

Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** Run the **ospf memory-overload control disable** command to disable OSPF memory overload control.

 NOTE

To reduce the impact of memory overload on services, you are advised not to disable OSPF memory overload control.

- Step 3** Run the **commit** command to commit the configuration.

----End

Disabling OSPF CPU Overload Control

The OSPF CPU overload control function is enabled by default. You can disable this function if necessary.

Context

By default, OSPF CPU overload control is enabled. If a device's CPU is overloaded, each module takes necessary measures to control its own CPU usage accordingly. Upon receiving a CPU overload notification from the system, the OSPF module controls the speeds of some internal computing processes and the establishment of neighbor relationships based on the CPU overload condition to enhance the resilience of OSPF. In this case, new neighbor relationships cannot be established. For original neighbor relationships, if a neighbor relationship is in the Full state, it will be retained; if a neighbor relationship is in a non-Full state, establishment of the neighbor relationship is paused and can continue only after the CPU recovers from overload.

Procedure

- Step 1** Run **system-view**

The system view is displayed.

- Step 2** Run **ospf cpu-overload control disable**

OSPF CPU overload control is disabled.

 NOTE

To minimize the impact of CPU overload upon services, you are advised not to disable OSPF CPU overload control.

Step 3 Run commit

The configuration is committed.

----End

1.1.4.2.27 Configuring OSPF Multi-Area Adjacency

OSPF multi-area adjacency allows one link to be shared by multiple OSPF areas.

Usage Scenario

According to OSPF, intra-area paths take precedence over inter-area paths. If a link in an area is a high-speed link, routes in other areas cannot use the high-speed link for transmission at the same time. Instead, they can use only low-speed links. To solve this problem, run the **ospf enable multi-area** command to enable OSPF on the multi-area adjacency interface so that a path can be shared by multiple areas.

Among the OSPF features supported by an OSPF multi-area adjacency interface, some can be inherited from the main interface, whereas others need to be configured independently. For details, see [Table 1-50](#).

Table 1-50 OSPF features supported by OSPF multi-area adjacency interfaces

OSPF Feature	Description	Related Configuration Links
BFD for OSPF	BFD for OSPF can be inherited from main interfaces, except for the configuration that disables BFD from an interface. This configuration needs to be configured independently.	Disabling an OSPF Multi-Area Adjacency Interface from Creating BFD Sessions
OSPF fast convergence	OSPF multi-area adjacency interfaces support OSPF fast convergence which needs to be configured independently for these interfaces and cannot be inherited from their main interfaces.	Configuring Fast Convergence for a Multi-Area Adjacency Interface

OSPF Feature	Description	Related Configuration Links
IGP-LDP synchronization	OSPF multi-area adjacency interfaces support IGP-LDP synchronization which needs to be configured independently for these interfaces and cannot be inherited from their main interfaces.	Configuring LDP-IGP Synchronization
OSPF IP FRR	OSPF multi-area adjacency interfaces support OSPF IP FRR. OSPF IP FRR can be inherited from main interfaces, except for the configuration that disables FRR from a specified OSPF interface. This configuration needs to be configured independently for OSPF multi-area adjacency interfaces.	Disabling OSPF IP FRR on an OSPF Multi-Area Adjacency Interface
OSPF neighbor relationship flapping suppression	OSPF multi-area adjacency interfaces support OSPF neighbor relationship flapping suppression which needs to be configured independently for these interfaces and cannot be inherited from their main interfaces.	Configuring Neighbor Relationship Flapping Suppression on an OSPF Multi-Area Adjacency Interface
OSPF flush LSA source tracing	OSPF multi-area adjacency interfaces support OSPF flush LSA source tracing which can be inherited from their main interfaces.	1.1.4.2.12 Configuring OSPF Flush Source Tracing
OSPF TE	OSPF multi-area adjacency interfaces inherit TE information from their main interfaces.	Configuring IGP TE (OSPF or IS-IS)

Pre-configuration Tasks

Before configuring OSPF multi-area adjacency, complete the following tasks:

- Configure a link layer protocol.
- Assign an IP address to each interface to ensure reachability between neighboring nodes at the network layer.
- [Configure basic OSPF functions.](#)

Enabling OSPF on a Multi-Area Adjacency Interface

Enabling OSPF on a multi-area adjacency interface is the prerequisite for configuring basic multi-area adjacency functions.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf [process-id | router-id router-id] ***

An OSPF process is started, and the OSPF view is displayed.

Step 3 Run **area area-id**

The OSPF area view is displayed.

Step 4 Enable OSPF on an interface.

1. Run twice **quit**

The system view is displayed.

2. Run **interface interface-type interface-number**

The interface view is displayed.

3. Run **ospf enable [process-id] area area-id**

OSPF is enabled on the interface.

Step 5 Run **ospf enable multi-area area-id**

OSPF is enabled on the multi-area adjacency interface.

Step 6 Run **ospf mtu-enable multi-area area-id**

The multi-area adjacency interface is configured to fill in DD packets with the MTU value when sending the packets and check whether the MTU in each DD packet received from a neighbor exceeds the MTU of the local interface.

NOTICE

Enabling an interface to fill in DD Packets with its MTU will cause the involved neighbor relationship to be re-established.

Step 7 Run **commit**

The configuration is committed.

----End

Configuring a Cost for a Multi-Area Adjacency Interface

Configuring a cost for a multi-area adjacency interface can control route selection.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run interface *interface-type interface-number*

The interface view is displayed.

Step 3 Run ospf enable [*process-id*] area *area-id*

OSPF is enabled on the interface.

Step 4 Run ospf enable multi-area *area-id*

OSPF is enabled on the multi-area adjacency interface.

Step 5 Run ospf cost *cost* multi-area *area-id*

A cost is configured for the multi-area adjacency interface.

Step 6 Run commit

The configuration is committed.

----End

Configuring an Authentication Mode for a Multi-Area Adjacency Interface

Configuring an authentication mode for a multi-area adjacency interface improves OSPF network security.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run interface *interface-type interface-number*

The interface view is displayed.

Step 3 Run ospf enable [*process-id*] area *area-id*

OSPF is enabled on the interface.

Step 4 Run ospf enable multi-area *area-id*

OSPF is enabled on the multi-area adjacency interface.

Step 5 Run either of the following commands:

- Run the **ospf authentication-mode simple [plain *plain-text* | [cipher] *cipher-text*] multi-area *area-id*** command to configure simple authentication for the multi-area adjacency interface.
- Run the **ospf authentication-mode { md5 | hmac-md5 | hmac-sha256 } [key-id { plain *plain-text* | [cipher] *cipher-text* }] multi-area *area-id*** command to configure ciphertext authentication for the multi-area adjacency interface.

- Run the **ospf authentication-mode keychain *keychain-name multi-area area-id*** command to configure keychain authentication for the multi-area adjacency interface.
- Run the **ospf authentication-mode null multi-area *area-id*** command to configure null authentication for the multi-area adjacency interface.

Step 6 Run **commit**

The configuration is committed.

----End

Configuring Fast Convergence for a Multi-Area Adjacency Interface

Configuring fast convergence for a multi-area adjacency interface improves network performance.

Procedure

- Configure the interval at which a multi-area adjacency interface sends Hello packets.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface *interface-type interface-number***
The OSPF interface view is displayed.
 - c. Run **ospf enable [*process-id*] area *area-id***
OSPF is enabled on the interface.
 - d. Run **ospf enable multi-area *area-id***
OSPF is enabled on the multi-area adjacency interface.
 - e. Run **ospf timer hello *interval multi-area area-id***
The interval at which Hello packets are sent is configured on the multi-area adjacency interface.
 - f. Run **commit**
The configuration is committed.
- Configure the dead interval of OSPF neighbor relationships for a multi-area adjacency interface.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface *interface-type interface-number***
The OSPF interface view is displayed.
 - c. Run **ospf enable [*process-id*] area *area-id***
OSPF is enabled on the interface.
 - d. Run **ospf enable multi-area *area-id***
OSPF is enabled on the multi-area adjacency interface.

- e. Run **ospf timer dead** *interval multi-area area-id*
The dead interval of OSPF neighbor relationships is configured for the multi-area adjacency interface.
- f. Run **commit**
The configuration is committed.
- Configure an LSA retransmission interval on a multi-area adjacency interface.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface** *interface-type interface-number*
The OSPF interface view is displayed.
 - c. Run **ospf enable** [*process-id*] **area** *area-id*
OSPF is enabled on the interface.
 - d. Run **ospf enable multi-area** *area-id*
OSPF is enabled on the multi-area adjacency interface.
 - e. Run **ospf timer retransmit** *interval multi-area area-id*
An LSA retransmission interval is configured on the multi-area adjacency interface.
 - f. Run **commit**
The configuration is committed.
- Configure an LSA transmission delay on a multi-area adjacency interface.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface** *interface-type interface-number*
The OSPF interface view is displayed.
 - c. Run **ospf enable** [*process-id*] **area** *area-id*
OSPF is enabled on the interface.
 - d. Run **ospf enable multi-area** *area-id*
OSPF is enabled on the multi-area adjacency interface.
 - e. Run **ospf trans-delay** *interval multi-area area-id*
An LSA transmission delay is configured on the multi-area adjacency interface.
 - f. Run **commit**
The configuration is committed.
- Configure smart-discover on a multi-area adjacency interface.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface** *interface-type interface-number*
The OSPF interface view is displayed.

- c. Run **ospf enable [process-id] area area-id**
OSPF is enabled on the interface.
 - d. Run **ospf enable multi-area area-id**
OSPF is enabled on the multi-area adjacency interface.
 - e. Run **ospf smart-discover multi-area area-id**
Smart-discover is enabled on the multi-area adjacency interface.
Without smart-discover, if the neighbor status of a device on a P2P network changes or the DR or BDR on a multi-access network (broadcast or NBMA network) changes, Hello packets are not sent to neighbors until the Hello timer expires. This slows down the establishment of neighbor relationships. With smart-discover, Hello packets are sent to neighbors immediately in this case, regardless of the Hello timer. This speeds up neighbor relationship establishment and network convergence.
 - f. Run **commit**
The configuration is committed.
- Configure an OSPF multi-area adjacency interface to filter the LSAs to be sent.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **interface interface-type interface-number**
The OSPF interface view is displayed.
 - c. Run **ospf enable [process-id] area area-id**
OSPF is enabled on the interface.
 - d. Run **ospf enable multi-area area-id**
OSPF is enabled on the multi-area adjacency interface.
 - e. Run **ospf filter-lsa-out { all | { summary [acl { acl-number| acl-name }] | ase [acl { acl-number| acl-name }] | nssa [acl { acl-number| acl-name }] } * } multi-area area-id**
The multi-area adjacency interface is configured to filter the LSAs to be sent.
If multiple links exist between two devices, you can configure the local device to filter the LSAs to be sent. This prevents unnecessary LSA transmission and reserves bandwidth resources.
 - f. Run **commit**
The configuration is committed.

----End

Configuring Neighbor Relationship Flapping Suppression on an OSPF Multi-Area Adjacency Interface

Neighbor relationship flapping suppression can be configured on an OSPF multi-area adjacency interface to delay OSPF neighbor relationship reestablishment or set the link cost to the maximum value in case of neighbor relationship flapping.

Usage Scenario

If an interface carrying OSPF services alternates between Up and Down, OSPF neighbor relationship flapping occurs on the interface. During the flapping, OSPF frequently sends Hello packets to reestablish the neighbor relationship, synchronizes LSDBs, and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, OSPF services, and other OSPF-dependent services, such as LDP and BGP. OSPF neighbor relationship flapping suppression can address this problem by delaying OSPF neighbor relationship reestablishment or preventing service traffic from passing through flapping links.



The following steps are optional, and choose them as required.

Procedure

Step 1 Run **system-view**

The system view is displayed.

To disable the function globally, run the **suppress-flapping peer disable** command.

Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

Step 3 Run **ospf enable [process-id] area area-id**

OSPF is enabled on the interface.

Step 4 Run **ospf enable multi-area area-id**

OSPF is enabled on the multi-area adjacency interface.

Step 5 (Optional) Run **ospf suppress-flapping peer disable multi-area area-id**

OSPF neighbor relationship flapping suppression on a multi-area adjacency interface is disabled.

To disable OSPF neighbor relationship flapping suppression from one of the interfaces, run this command.

Step 6 Run **ospf suppress-flapping peer hold-down interval multi-area area-id**

The Hold-down mode and its duration are configured for the multi-area adjacency interface.

OSPF neighbor relationship flapping suppression operates in Hold-down or Hold-max-cost mode.

- Hold-down mode: In the case of frequent flooding and topology changes during neighbor relationship establishment, interfaces prevent neighbor relationship reestablishment during the suppression period, which reduces LSDB synchronization attempts and packet exchanges.
- Hold-max-cost mode: If the traffic forwarding path changes frequently, interfaces use 65535 (maximum value) as the cost of the flapping link during the suppression period. This prevents traffic from passing through the flapping link.

 NOTE

To change the value of Max-cost for OSPF links, run the **maximum-link-cost cost** command.

The Hold-down mode and Hold-max-cost mode can be both used. When both modes are configured, the device first enters the Hold-down mode. After exiting the Hold-down mode, the device enters the Hold-max-cost mode.

To disable the Hold-max-cost mode, run the **ospf suppress-flapping peer hold-max-cost disable multi-area** command.

Step 7 Run **ospf suppress-flapping peer { detecting-interval *detecting-interval* | threshold *threshold* | resume-interval *resume-interval* } * multi-area *area-id***

Detection parameters are configured for OSPF neighbor relationship flapping suppression on the multi-area adjacency interface.

- Configure a threshold for exiting OSPF neighbor relationship flapping suppression.
If the interval between two successive neighbor relationship state changes from Full to a non-Full state is longer than the *resume-interval*, the flapping_count is reset.
- If OSPF neighbor relationship flapping suppression works in Hold-max-cost mode, the value of *resume-interval* indicates the duration of this mode.

 NOTE

The value of *resume-interval* must be greater than that of *detecting-interval*.

You can configure detection parameters for OSPF neighbor relationship flapping suppression on specific interfaces according to network conditions. However, using the default values of these parameters is recommended.

Step 8 Run **quit**

The system view is displayed.

Step 9 Run **quit**

The user view is displayed.

Step 10 Run **reset ospf suppress-flapping process-id peer [interface-name [all-areas | area *area-id*] | interface-type *interface-number* [all-areas | area*area-id*]] [notify-peer]**

The OSPF multi-area adjacency interface is configured to exit OSPF neighbor relationship flapping suppression.

 NOTE

The interface exits flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding OSPF process is reset.
- An OSPF neighbor is reset using the **reset ospf peer** command.
- OSPF neighbor relationship flapping suppression is disabled globally using the **suppress-flapping peer disable (OSPF)** command.
- The **reset ospf suppress-flapping peer** command is run.

Step 11 Run commit

The configuration is committed.

----End

Disabling an OSPF Multi-Area Adjacency Interface from Creating BFD Sessions

After BFD for OSPF is configured, all interfaces on which neighbor relationships are in Full state in the OSPF process create BFD sessions. You can disable an interface from creating BFD sessions as required.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run `interface interface-type interface-number`

The interface view is displayed.

Step 3 Run `ospf enable [process-id] area area-id`

OSPF is enabled on the interface.

Step 4 Run `ospf enable multi-area area-id`

OSPF is enabled on the multi-area adjacency interface.

Step 5 Run `ospf bfd block multi-area area-id`

The OSPF multi-area adjacency interface is disabled from creating BFD sessions.

Step 6 Run commit

The configuration is committed.

----End

Disabling OSPF IP FRR on an OSPF Multi-Area Adjacency Interface

OSPF IP fast reroute (FRR) can be disabled on an OSPF multi-area adjacency interface that is connected to a link carrying key services. This prevents FRR from calculating the link as the backup link and minimizes any potential impact on the services.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

Step 3 Run **ospf enable [process-id] area area-id**

OSPF is enabled on the interface.

Step 4 Run **ospf enable multi-area area-id**

OSPF is enabled on the multi-area adjacency interface.

Step 5 Run **ospf frr block multi-area area-id**

OSPF IP FRR is disabled on the OSPF multi-area adjacency interface.

Step 6 (Optional) Run **ospf remote-lfa disable multi-area area-id**

Remote loop-free alternate (LFA) is disabled on the OSPF multi-area adjacency interface.

Step 7 Run **commit**

The configuration is committed.

----End

Configuring a Fallback Cost for an Eth-Trunk Multi-Area Adjacency Interface

Configuring a fallback cost for an Eth-Trunk multi-area adjacency interface helps control route selection.

Context

After an Eth-trunk member interface goes down, the OSPF cost value is automatically adjusted. When an Eth-trunk member interface becomes invalid, the remaining bandwidth may fail to meet user requirements, causing service loss. In this fault scenario, the cost of the Eth-Trunk can be dynamically adjusted to a larger value so that traffic is transmitted through other paths. When the interface bandwidth is less than the fallback bandwidth threshold, the device changes the interface cost to the configured fallback cost in time so that a better transmission path is selected. When the bandwidth of an Eth-Trunk interface is greater than or equal to the configured fallback bandwidth threshold, cost-fallback does not take effect.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **interface eth-trunk trunk-id**

The Eth-Trunk interface view is displayed.

Step 3 Run **ospf enable [process-id] area area-id**

OSPF is enabled on the interface.

Step 4 Run **ospf enable multi-area area-id**

OSPF is enabled on the multi-area adjacency interface.

Step 5 Run **ospf cost-fallback fallbackcost threshold fallbackbw multi-area area-id**

A fallback cost is configured for an Eth-Trunk multi-area adjacency interface.

Step 6 Run **commit**

The configuration is committed.

----End

Maintaining OSPF on a Multi-Area Adjacency Interface

Context

NOTICE

OSPF statistics cannot be restored after being cleared. Therefore, exercise caution before clearing the OSPF statistics.

To clear OSPF statistics, run the following command in the user view.

Procedure

- Run **reset ospf [process-id] counters [neighbor [interface-type interface-number [all-areas | area area-id]] | [interface-name [all-areas | area area-id]] [router-id]]**

OSPF statistics are cleared.

----End

Verifying the Configuration of OSPF Multi-Area Adjacency

After configuring OSPF multi-area adjacency, verify the configuration.

Prerequisites

OSPF multi-area adjacency has been configured.

Procedure

- Run the following commands and check **(M) Indicates MADJ interface**, **Multi-area interface**, and **Multi-area Interface Count** fields to view configurations on the multi-area adjacency interface:
 - display ospf brief**
 - display ospf interface**

– display ospf peer
----End

1.1.4.2.28 Configuration Examples for OSPF

This section provides several OSPF configuration examples.

Example for Configuring Basic OSPF Functions

This section describes how to configure basic OSPF functions, including enabling OSPF on each router and specifying network segments in different areas.

Networking Requirements

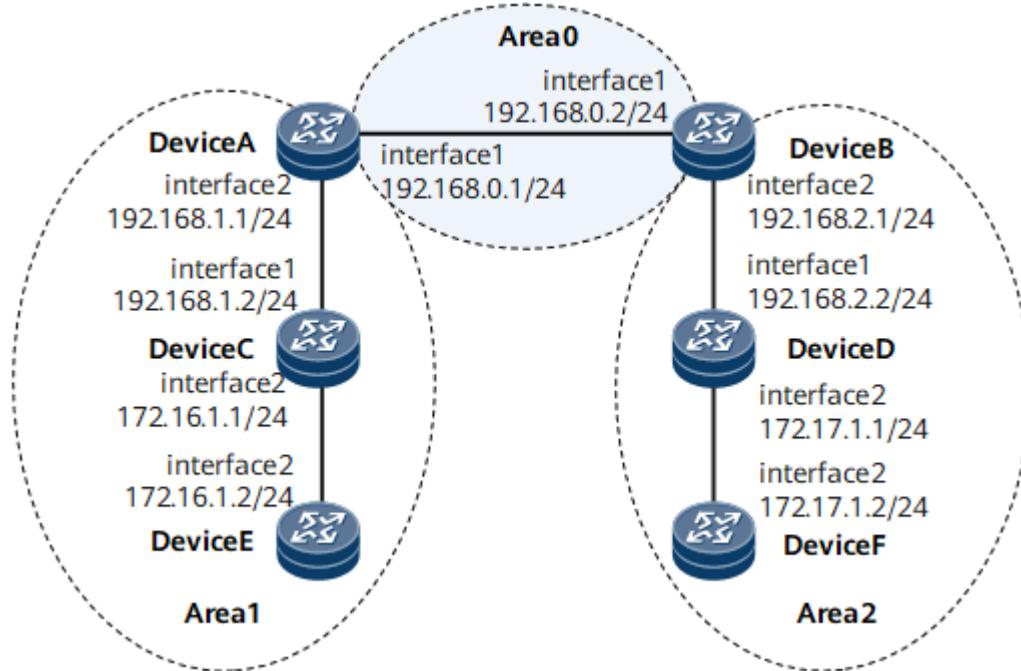
On the network shown in [Figure 1-126](#), all routers run OSPF, and the entire AS is divided into three areas. DeviceA and DeviceB function as ABRs to forward inter-area routes.

After the configuration is complete, each router should learn the routes to all network segments in the AS.

Figure 1-126 Networking for configuring basic OSPF functions



Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Precautions

When configuring basic OSPF functions, note the following rules:

- The backbone area is responsible for forwarding inter-area routes. In addition, the routing information between non-backbone areas must be forwarded through the backbone area. OSPF defines the following rules for the backbone area:
 - Connectivity must be available between non-backbone areas and the backbone area.
 - Connectivity must be available over the backbone area.
- The intervals at which Hello, Dead, and Poll packets are sent on the local interface must be the same as those on the remote interface. Otherwise, the OSPF neighbor relationship cannot be established.

Configuration Roadmap

The configuration roadmap is as follows:

1. Enable OSPF on each router.
2. Specify network segments in different areas.
3. Configure ciphertext authentication for the OSPF area.

Data Preparation

To complete the configuration, you need the following data:

Device Name	Router ID	Process ID	IP Address
Device A	1.1.1.1	1	Area 0: 192.168.0.0/24 Area 1: 192.168.1.0/24
Device B	2.2.2.2	1	Area 0: 192.168.0.0/24 Area 2: 192.168.2.0/24
Device C	3.3.3.3	1	Area 1: 192.168.1.0/24 and 172.16.1.0/24
Device D	4.4.4.4	1	Area 2: 192.168.2.0/24 and 172.17.1.0/24
Device E	5.5.5.5	1	Area 1: 172.16.1.0/24

Device F	6.6.6.6	1	Area 2: 172.17.1.0/24
----------	---------	---	--------------------------

Procedure

Step 1 Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic OSPF functions.

Configure DeviceA.

```
[~DeviceA] router id 1.1.1.1
[*DeviceA] ospf 1
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0] network 192.168.0.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0] quit
[*DeviceA-ospf-1] area 1
[*DeviceA-ospf-1-area-0.0.1] network 192.168.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.1] quit
[*DeviceA-ospf-1] commit
```

Configure DeviceB.

```
[~DeviceB] router id 2.2.2.2
[*DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0] network 192.168.0.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0] quit
[*DeviceB-ospf-1] area 2
[*DeviceB-ospf-1-area-0.0.2] network 192.168.2.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.2] quit
[*DeviceB-ospf-1] commit
```

Configure DeviceC.

```
[~DeviceC] router id 3.3.3.3
[*DeviceC] ospf 1
[*DeviceC-ospf-1] area 1
[*DeviceC-ospf-1-area-0.0.1] network 192.168.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.1] network 172.16.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.1] commit
[~DeviceC-ospf-1-area-0.0.1] quit
```

Configure DeviceD.

```
[~DeviceD] router id 4.4.4.4
[*DeviceD] ospf 1
[*DeviceD-ospf-1] area 2
[*DeviceD-ospf-1-area-0.0.2] network 192.168.2.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.2] network 172.17.1.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.2] commit
[~DeviceD-ospf-1-area-0.0.2] quit
```

Configure DeviceE.

```
[~DeviceE] router id 5.5.5.5
[*DeviceE] ospf 1
[*DeviceE-ospf-1] area 1
[*DeviceE-ospf-1-area-0.0.1] network 172.16.1.0 0.0.0.255
[*DeviceE-ospf-1-area-0.0.1] commit
[~DeviceE-ospf-1-area-0.0.1] quit
```

Configure DeviceF.

```
[~DeviceF] router id 6.6.6.6
```

```
[~DeviceF] ospf 1
[*DeviceF-ospf-1] area 2
[*DeviceF-ospf-1-area-0.0.0.2] network 172.17.1.0 0.0.0.255
[*DeviceF-ospf-1-area-0.0.0.2] commit
[~DeviceF-ospf-1-area-0.0.0.2] quit
```

Step 3 Configure ciphertext authentication for the OSPF area.

Configure DeviceA.

```
[~DeviceA] ospf 1
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0.0] authentication-mode hmac-sha256 1 cipher YsHsjx_202206
[*DeviceA-ospf-1-area-0.0.0.0] quit
[*DeviceA-ospf-1] commit
```

Configure DeviceB.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] authentication-mode hmac-sha256 1 cipher YsHsjx_202206
[*DeviceB-ospf-1-area-0.0.0.0] quit
[*DeviceB-ospf-1] commit
```



Device B and Device A must be configured with the same password. Otherwise, the neighbor relationship cannot be established.

Step 4 Verify the configuration.

Display the OSPF neighbors of DeviceA.

```
[~DeviceA] display ospf peer
(M) Indicates MADJ neighbor
      OSPF Process 1 with Router ID 1.1.1.1
      Neighbors

Area 0.0.0.0 interface 192.168.0.1(GigabitEthernet1/0/0)'s neighbors
Router ID: 2.2.2.2    Address: 192.168.0.2
State: Full Mode:Nbr is Master Priority: 1
DR: 192.168.0.2    BDR: 192.168.0.1    MTU: 0
Dead timer due in 36 sec
Retrans timer interval: 5
Neighbor is up for 1h15m4s
Neighbor up time : 2020-06-08 01:41:57
Authentication Sequence: [ 0 ]

Area 0.0.0.1 interface 192.168.1.1(GigabitEthernet2/0/0)'s neighbors
Router ID: 3.3.3.3    Address: 192.168.1.2
State: Full Mode:Nbr is Master Priority: 1
DR: 192.168.1.2    BDR: 192.168.1.1    MTU: 0
Dead timer due in 39 sec
Retrans timer interval: 5
Neighbor is up for 1h15m4s
Neighbor up time : 2020-06-08 01:41:57
Authentication Sequence: [ 0 ]
```

Display the OSPF routes of DeviceA.

```
[~DeviceA] display ospf routing
      OSPF Process 1 with Router ID 1.1.1.1
      Routing Tables

      Routing for Network
      Destination    Cost     Type    NextHop      AdvRouter      Area
      172.16.1.0/24   2        Transit   192.168.1.2   3.3.3.3      0.0.0.1
      172.17.1.0/24   3        Inter-area 192.168.0.2   2.2.2.2      0.0.0.0
      192.168.2.0/24   2        Inter-area 192.168.0.2   2.2.2.2      0.0.0.0
```

```
Total Nets: 3
Intra Area: 1 Inter Area: 2 ASE: 0 NSSA: 0
```

Display the LSDB of DeviceA.

```
[~DeviceA] display ospf lsdb
OSPF Process 1 with Router ID 1.1.1.1
Link State Database

Area: 0.0.0.0
Type LinkState ID AdvRouter Age Len Sequence Metric
Router 1.1.1.1 1.1.1.1 93 48 80000004 1
Router 2.2.2.2 2.2.2.2 92 48 80000004 1
Sum-Net 172.16.1.0 1.1.1.1 1287 28 80000002 2
Sum-Net 192.168.1.0 1.1.1.1 1716 28 80000001 1
Sum-Net 172.17.1.0 2.2.2.2 1336 28 80000001 2
Sum-Net 192.168.2.0 2.2.2.2 87 28 80000002 1

Area: 0.0.0.1
Type LinkState ID AdvRouter Age Len Sequence Metric
Router 1.1.1.1 1.1.1.1 1420 48 80000002 1
Router 3.3.3.3 3.3.3.3 1294 60 80000003 1
Router 5.5.5.5 5.5.5.5 1296 36 80000002 1
Network 172.16.1.1 3.3.3.3 1294 32 80000001 0
Sum-Net 172.17.1.0 1.1.1.1 1325 28 80000001 3
Sum-Net 192.168.0.0 1.1.1.1 1717 28 80000001 1
Sum-Net 192.168.2.0 1.1.1.1 1717 28 80000001 2
```

Display the routing table on DeviceD and perform the ping operation to test the connectivity.

```
[~DeviceD] display ospf routing
OSPF Process 1 with Router ID 4.4.4.4
Routing Tables

Routing for Network
Destination Cost Type NextHop AdvRouter Area
172.16.1.0/24 4 Inter-area 192.168.2.1 2.2.2.2 0.0.0.2
192.168.0.0/24 2 Inter-area 192.168.2.1 2.2.2.2 0.0.0.2
192.168.1.0/24 3 Inter-area 192.168.2.1 2.2.2.2 0.0.0.2

Total Nets: 3
Intra Area: 0 Inter Area: 3 ASE: 0 NSSA: 0
[~DeviceD] ping 172.16.1.1
PING 172.16.1.1: 56 data bytes, press CTRL_C to break
Reply from 172.16.1.1: bytes=56 Sequence=1 ttl=253 time=62 ms
Reply from 172.16.1.1: bytes=56 Sequence=2 ttl=253 time=16 ms
Reply from 172.16.1.1: bytes=56 Sequence=3 ttl=253 time=62 ms
Reply from 172.16.1.1: bytes=56 Sequence=4 ttl=253 time=94 ms
Reply from 172.16.1.1: bytes=56 Sequence=5 ttl=253 time=63 ms

--- 172.16.1.1 ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 16/59/94 ms
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
ospf 1
area 0.0.0
network 192.168.0.0 0.0.0.255
authentication-mode hmac-sha256 1 cipher %^%#c;\wJ4Qi8l1FMGM}KmlK9rha/.D.!$"~0(Ep66z~%^%
%#
area 0.0.1
network 192.168.1.0 0.0.0.255
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
router id 2.2.2.2
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.1 255.255.255.0
#
ospf 1
area 0.0.0
network 192.168.0.0 0.0.0.255
authentication-mode hmac-sha256 1 cipher %^%#c;\wJ4Qi8l1FMGM}KmlK9rha/.D.!$"~0(Ep66z~%^%
%#
area 0.0.2
network 192.168.2.0 0.0.0.255
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
ospf 1
area 0.0.1
network 192.168.1.0 0.0.0.255
network 172.16.1.0 0.0.0.255
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
router id 4.4.4.4
#
interface GigabitEthernet1/0/0
```

```
undo shutdown
ip address 192.168.2.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.17.1.1 255.255.255.0
#
ospf 1
area 0.0.0.2
network 192.168.2.0 0.0.0.255
network 172.17.1.0 0.0.0.255
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
router id 5.5.5.5
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.2 255.255.255.0
#
ospf 1
area 0.0.0.1
network 172.16.1.0 0.0.0.255
#
return
```

- DeviceF configuration file

```
#
sysname DeviceF
#
router id 6.6.6.6
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.17.1.2 255.255.255.0
#
ospf 1
area 0.0.0.2
network 172.17.1.0 0.0.0.255
#
return
```

Example for Configuring OSPF Virtual Links

This section describes how to configure virtual links to connect a non-backbone area to the backbone area.

Networking Requirements

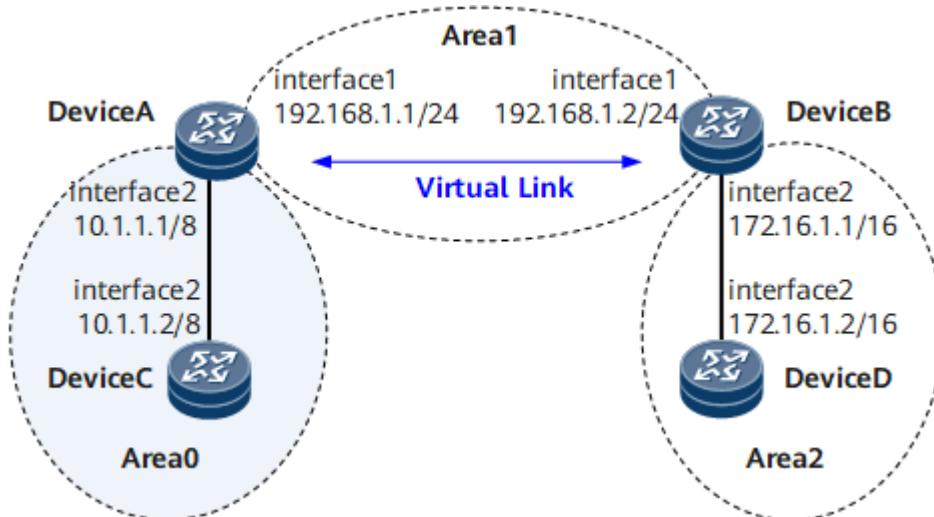
On the network shown in [Figure 1-127](#), Area 2 is not directly connected to the backbone area Area 0. Area 1 serves as a transit area to connect Area 2 and Area 0. A virtual link is configured between DeviceA and DeviceB.

Figure 1-127 Networking for configuring an OSPF virtual links



NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Precautions

The default value is recommended when a virtual link is created. You can modify the value in actual scenarios.

Suggestions for configuring parameters are as follows:

- The smaller the **hello** parameter, the more rapidly a router detects network topology change, the more network resources are consumed.
- If the **retransmit** parameter is set too small, LSAs may be retransmitted. Setting the parameter to a large value is recommended in a low-speed network.
- The authentication modes of a virtual link and the backbone area must be the same.
- To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on each router.
2. Configure a virtual link between DeviceA and DeviceB to connect a non-backbone area to the backbone area.

Data Preparation

To complete the configuration, you need the following data:

Device Name	Router ID	Process ID	IP Address
-------------	-----------	------------	------------

DeviceA	1.1.1.1	1	Area 0: 10.0.0.0/8 Area 1: 192.168.1.0/24
DeviceB	2.2.2.2	1	Area 1: 192.168.1.0/24 Area 2: 172.16.0.0/16
DeviceC	3.3.3.3	1	Area 0: 10.0.0.0/8
DeviceD	4.4.4.4	1	Area 2: 172.16.0.0/16

Procedure

Step 1 Assign an IP address to each interface according to [Configuration Files](#). For configuration details, see Configuration Files in this section.

Step 2 Configure basic OSPF functions.

[Configuring Basic OSPF Functions](#) shows how to configure basic OSPF functions. For configuration details, see Configuration Files in this section.

Step 3 Check the OSPF routes on DeviceA

```
[~DeviceA] display ospf routing
      OSPF Process 1 with Router ID 1.1.1.1
      Routing Tables

      Routing for Network
      Destination    Cost     Type    NextHop      AdvRouter      Area
      10.0.0.0/8      1       Transit   10.1.1.1    3.3.3.3      0.0.0.0
      192.168.1.0/24  1       Transit   192.168.1.1  1.1.1.1      0.0.0.1

      Total Nets: 2
      Intra Area: 2 Inter Area: 0 ASE: 0 NSSA: 0
```

The routing table on Device A contains no route in Area 2 because Area 2 is not directly connected to Area 0.

Step 4 Configure an OSPF virtual link.

Configure DeviceA.

```
[~DeviceA] router id 1.1.1.1
[~DeviceA] ospf 1
[*DeviceA-ospf-1] area 1
[*DeviceA-ospf-1-area-0.0.0.1] vlink-peer 2.2.2.2
[*DeviceA-ospf-1-area-0.0.0.1] quit
[*DeviceA-ospf-1] quit
[*DeviceA] commit
```

Configure DeviceB.

```
[~DeviceB] router id 2.2.2.2
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 1
```

```
[*DeviceB-ospf-1-area-0.0.0.1] vlink-peer 1.1.1.1
[*DeviceB-ospf-1-area-0.0.0.1] quit
[*DeviceB-ospf-1] quit
[*DeviceB] commit
```

Step 5 Verify the configuration.

Display the OSPF vlink on DeviceA.

```
[~DeviceA] display ospf vlink
    OSPF Process 1 with Router ID 1.1.1.1
        Virtual Links
    Virtual-link Neighbor-id -> 2.2.2.2, Neighbor-State: Full
    Interface: 192.168.1.1 (GigabitEthernet1/0/0)
    Cost: 1 State: P-2-P Type: Virtual
    Transit Area: 0.0.0.1
    Timers: Hello 10 , Dead 40 , Retransmit 5 , Transmit Delay 1
    GR State: Normal
```

The preceding command output shows that the OSPF vlink neighbor status is "Full".

Display the OSPF routes on DeviceA.

```
[~DeviceA] display ospf routing
    OSPF Process 1 with Router ID 1.1.1.1
        Routing Tables

    Routing for Network
    Destination      Cost      Type      Nexthop      AdvRouter      Area
    172.16.0.0/16    2        Inter-area 192.168.1.2  2.2.2.2      0.0.0.2
    10.0.0.0/8       1        Transit    10.1.1.1    1.1.1.1      0.0.0.0
    192.168.1.0/24   1        Transit    192.168.1.1  1.1.1.1      0.0.0.1

    Total Nets: 3
    Intra Area: 2 Inter Area: 1 ASE: 0 NSSA: 0
```

After the virtual link is configured, the routing table on DeviceA contains routes in Area 2.

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.1 255.0.0.0
#
ospf 1
area 0.0.0.0
network 10.0.0.0 0.255.255.255
area 0.0.0.1
network 192.168.1.0 0.0.0.255
vlink-peer 2.2.2.2
#
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
router id 2.2.2.2  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.1.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.16.1.1 255.255.0.0  
#  
ospf 1  
area 0.0.0.1  
network 192.168.1.0 0.0.0.255  
vlink-peer 1.1.1.1  
area 0.0.0.2  
network 172.16.0.0 0.0.255.255  
#  
return
```

- DeviceC configuration file

```
#  
sysname DeviceC  
#  
router id 3.3.3.3  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.1.2 255.0.0.0  
#  
ospf 1  
area 0.0.0.0  
network 10.0.0.0 0.255.255.255  
#  
return
```

- DeviceD configuration file

```
#  
sysname DeviceD  
#  
router id 4.4.4.4  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.16.1.2 255.255.0.0  
#  
ospf 1  
area 0.0.0.2  
network 172.16.0.0 0.0.255.255  
#  
return
```

Example for Configuring an OSPF Stub Area

This section describes how to configure a stub area that imports static routes. Such configuration can reduce the number of LSAs advertised to this area without affecting route reachability.

Networking Requirements

On the network shown in [Figure 1-128](#), all routers run OSPF, and the entire AS is divided into three areas. DeviceA and DeviceB function as ABRs to advertise inter-

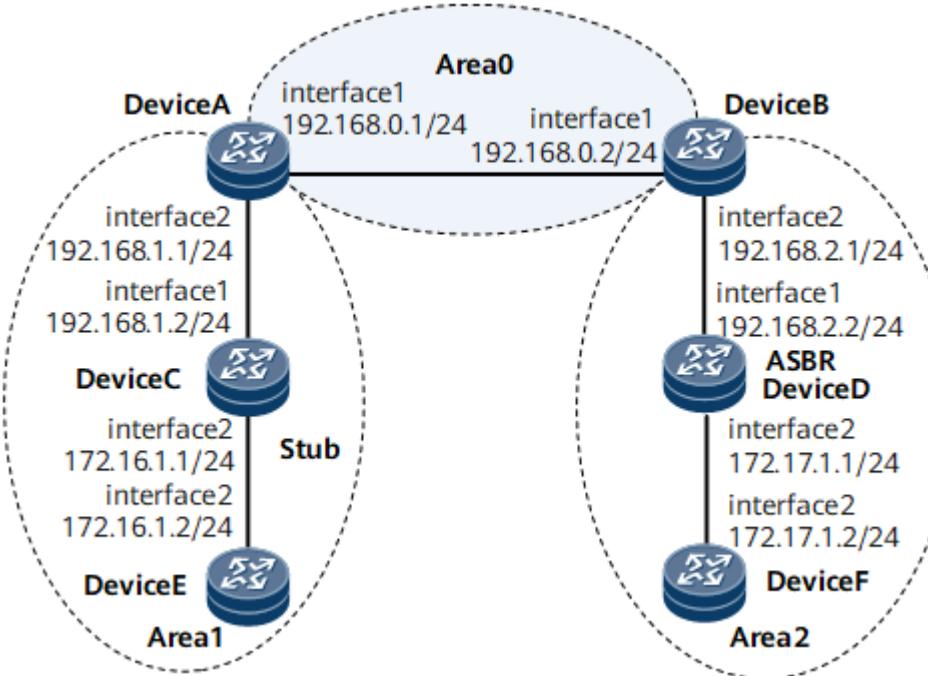
area routes; DeviceD functions as the ASBR and imports external routes (static routes).

Configure Area 1 as a stub area to reduce the LSAs advertised to this area without affecting route reachability.

Figure 1-128 Networking for configuring an OSPF stub area

 NOTE

Interfaces 1 and 2 in this example represent GE1/0/0 and GE2/0/0, respectively.



Precautions

When configuring an OSPF stub area, note the following rules:

- The backbone area cannot be configured as a stub area.
- An ASBR cannot exist in a stub area. That is, AS-external routes are not flooded in the stub area.
- A virtual link cannot pass through a stub area.
- To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on each router for interconnection.

2. Configure a static route on DeviceD and import it into OSPF.
3. Configure Area 1 as a stub area by running the **stub** command on all routers in Area 1 and check the OSPF routing information on DeviceC.
4. Prevent DeviceA from advertising Type 3 LSAs to the stub area, and check the OSPF routing information on DeviceC.

Data Preparation

To complete the configuration, you need the following data:

Device Name	Router ID	Process ID	IP Address
DeviceA	1.1.1.1	1	Area 0: 192.168.0.0/24 Area 1: 192.168.1.0/24
DeviceB	2.2.2.2	1	Area 0: 192.168.0.0/24 Area 2: 192.168.2.0/24
DeviceC	3.3.3.3	1	Area 1: 192.168.1.0/24 and 172.16.1.0/24
DeviceD	4.4.4.4	1	Area 2: 192.168.2.0/24 and 172.17.1.0/24
DeviceE	5.5.5.5	1	Area 1: 172.16.1.0/24
DeviceF	6.6.6.6	1	Area 2: 172.17.1.0/24

Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic OSPF functions. For details, see [Example for Configuring Basic OSPF Functions](#).
- Step 3** Configure DeviceD to import static routes.

```
[*DeviceD] ip route-static 10.0.0.0 8 null 0
[*DeviceD] ospf 1
[*DeviceD-ospf-1] import-route static type 1
[*DeviceD-ospf-1] commit
```

```
[~DeviceD-ospf-1] quit  
# Display the ABR and ASBR information on DeviceC.
```

```
[~DeviceC] display ospf abr-asbr  
OSPF Process 1 with Router ID 3.3.3.3  
Routing Table to ABR and ASBR  
Type Destination Area Cost NextHop RtType  
Intra-area 1.1.1.1 0.0.1 1 192.168.1.1 ABR  
Inter-area 4.4.4.4 0.0.1 3 192.168.1.1 ASBR
```

```
# Display the OSPF routing table on DeviceC.
```

NOTE

If the area where DeviceC resides is a common area, external routes exist in the routing table.

```
[~DeviceC] display ospf routing  
OSPF Process 1 with Router ID 3.3.3.3  
Routing Tables  
  
Routing for Network  
Destination Cost Type NextHop AdvRouter Area  
172.17.1.0/24 4 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1  
192.168.0.0/24 2 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1  
192.168.2.0/24 3 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1  
  
Routing for ASEs  
Destination Cost Type Tag NextHop AdvRouter  
10.0.0.0/8 4 Type1 1 192.168.1.1 4.4.4.4  
  
Total Nets: 4  
Intra Area: 0 Inter Area: 3 ASE: 1 NSSA: 0
```

Step 4 Configure Area 1 as a stub area.

```
# Configure DeviceA.
```

```
[~DeviceA] ospf 1  
[*DeviceA-ospf-1] area 1  
[*DeviceA-ospf-1-area-0.0.0.1] stub  
[*DeviceA-ospf-1-area-0.0.0.1] commit  
[~DeviceA-ospf-1-area-0.0.0.1] quit
```

```
# Configure DeviceC.
```

```
[~DeviceC] ospf 1  
[*DeviceC-ospf-1] area 1  
[*DeviceC-ospf-1-area-0.0.0.1] stub  
[*DeviceC-ospf-1-area-0.0.0.1] commit  
[~DeviceC-ospf-1-area-0.0.0.1] quit
```

```
# Configure DeviceE.
```

```
[~DeviceE] ospf 1  
[*DeviceE-ospf-1] area 1  
[*DeviceE-ospf-1-area-0.0.0.1] stub  
[*DeviceE-ospf-1-area-0.0.0.1] commit  
[~DeviceE-ospf-1-area-0.0.0.1] quit
```

```
# Display the routing table on DeviceC.
```

 NOTE

After the area where DeviceC resides is configured as a stub area, a default route rather than AS external routes is displayed in the routing table.

```
[~DeviceC] display ospf routing
OSPF Process 1 with Router ID 3.3.3.3
    Routing Tables

Routing for Network
Destination Cost Type NextHop AdvRouter Area
0.0.0.0/0 2 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1
172.17.1.0/24 4 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1
192.168.0.0/24 2 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1
192.168.2.0/24 3 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1

Total Nets: 4
Intra Area: 0 Inter Area: 4 ASE: 0 NSSA: 0
```

Step 5 # Prevent DeviceA from advertising Type 3 LSAs to the stub area.

```
[~DeviceA] ospf
[*DeviceA-ospf-1] area 1
[*DeviceA-ospf-1-area-0.0.0.1] stub no-summary
[*DeviceA-ospf-1-area-0.0.0.1] commit
[~DeviceA-ospf-1-area-0.0.0.1] quit
```

Step 6 Verify the configuration.

Display the OSPF routing table on DeviceC.

```
[~DeviceC] display ospf routing
OSPF Process 1 with Router ID 3.3.3.3
    Routing Tables

Routing for Network
Destination Cost Type NextHop AdvRouter Area
0.0.0.0/0 2 Inter-area 192.168.1.1 1.1.1.1 0.0.0.1

Total Nets: 1
Intra Area: 0 Inter Area: 1 ASE: 0 NSSA: 0
```

 NOTE

After the advertisement of summary LSAs to the stub area is disabled, the number of routing entries on the routers in the stub area further decreases, and only the default route to a destination beyond the stub area is reserved.

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
ospf 1
area 0.0.0.0
```

```
network 192.168.0.0 0.0.0.255
area 0.0.0.1
network 192.168.1.0 0.0.0.255
stub no-summary
#
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
router id 2.2.2.2  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.0.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.2.1 255.255.255.0  
#  
ospf 1  
area 0.0.0.0  
network 192.168.0.0 0.0.0.255  
area 0.0.0.2  
network 192.168.2.0 0.0.0.255  
#  
return
```

- DeviceC configuration file

```
#  
sysname DeviceC  
#  
router id 3.3.3.3  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.1.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.16.1.1 255.255.255.0  
#  
ospf 1  
area 0.0.0.1  
network 192.168.1.0 0.0.0.255  
network 172.16.1.0 0.0.0.255  
stub  
#  
return
```

- DeviceD configuration file

```
#  
sysname DeviceD  
#  
router id 4.4.4.4  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.2.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.17.1.1 255.255.255.0  
#  
ospf 1  
import-route static type 1  
area 0.0.0.2  
network 192.168.2.0 0.0.0.255  
network 172.17.1.0 0.0.0.255
```

```
#  
ip route-static 10.0.0.0 255.0.0.0 NULL0  
#  
return
```

- DeviceE configuration file

```
#  
sysname DeviceE  
#  
router id 5.5.5.5  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.16.1.2 255.255.255.0  
#  
ospf 1  
area 0.0.0.1  
network 172.16.1.0 0.0.0.255  
stub  
#  
return
```

- DeviceF configuration file

```
#  
sysname DeviceF  
#  
router id 6.6.6.6  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.17.1.2 255.255.255.0  
#  
ospf 1  
area 0.0.0.2  
network 172.17.1.0 0.0.0.255  
#  
return
```

Example for Configuring an OSPF NSSA

This section describes how to configure an OSPF not-so-stubby area (NSSA).

Networking Requirements

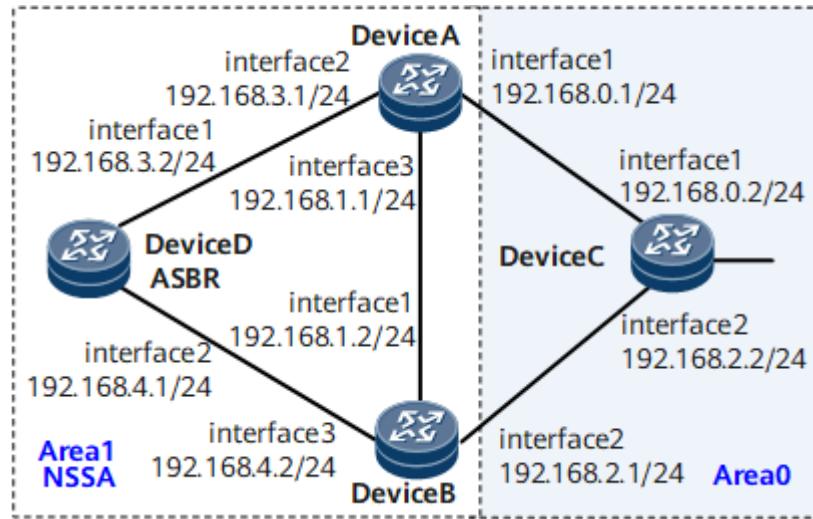
An excessive number of entries in a routing table wastes network resources and causes high CPU usage. To solve this problem, a non-backbone area on the border of an AS can be configured as an NSSA to reduce the amount of routing information to be transmitted. An NSSA in an AS does not transmit routes learned from other areas in the AS but imports AS external routes. This reduces bandwidth and storage resource consumption.

On the network shown in [Figure 1-129](#), all routers run OSPF and the entire AS is divided into two areas. DeviceA and DeviceB function as Area Border Routers (ABRs) to forward inter-area routes; DeviceD functions as an Autonomous System Boundary Router (ASBR) and imports the external static route 10.0.0.0/8. To import AS-external routes but reduce the number of Link State Advertisements (LSAs) advertised to area 1 without affecting route reachability, configure area 1 as an NSSA and configure DeviceA as an LSA translator in the NSSA.

Figure 1-129 Networking for configuring an OSPF NSSA

 NOTE

Interfaces 1 through 3 in this example represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

Configuration Roadmap

The configuration roadmap is as follows:

1. Enable OSPF on all routers and configure basic OSPF functions to ensure that routers communicate with each other using OSPF. For details, see [1.1.4.2.3 Configuring Basic OSPF Functions](#).
2. Configure area 1 as an NSSA.
3. Configure DeviceD to import the static route 10.0.0.0/8.
4. Configure DeviceA in the NSSA as an LSA translator.

Data Preparation

To complete the configuration, you need the following data:

- Router ID 1.1.1.1 of DeviceA; OSPF process ID 1; network segment 192.168.0.0/24 of area 0; network segments 192.168.1.0/24 and 192.168.3.0/24 of area 1
- Router ID 2.2.2.2 of DeviceB; OSPF process ID 1; network segment 192.168.2.0/24 of area 0; network segments 192.168.1.0/24 and 192.168.4.0/24 of area 1

- Router ID 3.3.3.3 of DeviceC; OSPF process ID 1; network segments 192.168.0.0/24 and 192.168.2.0/24 of area 0
- Router ID 4.4.4.4 of DeviceD; OSPF process ID 1; network segments 192.168.3.0/24 and 192.168.4.0/24 of area 1

Procedure

Step 1 Assign an IP address to each interface.

Configure an IP address for each interface as shown in [Figure 1-129](#). For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic OSPF functions.

[1.1.4.2.3 Configuring Basic OSPF Functions](#) shows how to configure basic OSPF functions. For details about the configuration, see [Configuration Files](#) in this section.

Step 3 Configure area 1 as an NSSA.

Configure DeviceA.

```
[~DeviceA] ospf  
[*DeviceA-ospf-1] area 1  
[*DeviceA-ospf-1-area-0.0.0.1] nssa  
[*DeviceA-ospf-1-area-0.0.0.1] commit  
[~DeviceA-ospf-1-area-0.0.0.1] quit
```

Configure DeviceB.

```
[~DeviceB] ospf  
[*DeviceB-ospf-1] area 1  
[*DeviceB-ospf-1-area-0.0.0.1] nssa  
[*DeviceB-ospf-1-area-0.0.0.1] commit  
[~DeviceB-ospf-1-area-0.0.0.1] quit
```

Configure DeviceD.

```
[~DeviceD] ospf  
[*DeviceD-ospf-1] area 1  
[*DeviceD-ospf-1-area-0.0.0.1] nssa  
[*DeviceD-ospf-1-area-0.0.0.1] commit  
[~DeviceD-ospf-1-area-0.0.0.1] quit
```



NSSA attributes must be configured on all routers in the NSSA using the **nssa** command.

Step 4 Configure DeviceD to import the static route 10.0.0.0/8.

```
[~DeviceD] ip route-static 10.0.0.0 8 null 0  
[~DeviceD] ospf  
[*DeviceD-ospf-1] import-route static  
[*DeviceD-ospf-1] commit  
[~DeviceD-ospf-1] quit
```

Display the OSPF routing table on DeviceC.

```
[~DeviceC] display ospf routing  
OSPF Process 1 with Router ID 3.3.3.3  
      Routing Tables  
  
      Routing for Network  
      Destination   Cost  Type    NextHop      AdvRouter   Area  
      192.168.3.0/24  2     Inter-area 192.168.0.1  1.1.1.1    0.0.0.0  
      192.168.4.0/24  2     Inter-area 192.168.2.1  2.2.2.2    0.0.0.0
```

```
192.168.0.0/24 1 Stub 192.168.0.2 3.3.3.3 0.0.0.0
192.168.1.0/24 2 Inter-area 192.168.0.1 1.1.1.1 0.0.0.0
192.168.1.0/24 2 Inter-area 192.168.2.1 2.2.2.2 0.0.0.0
192.168.2.0/24 1 Stub 192.168.2.2 3.3.3.3 0.0.0.0

Routing for ASEs
Destination Cost Type Tag NextHop AdvRouter
10.0.0/8 1 Type2 1 192.168.2.1 2.2.2.2

Total Nets: 7
Intra Area: 2 Inter Area: 4 ASE: 1 NSSA: 0
```

The command output shows that the router ID of the router that advertised the AS external route imported to the NSSA is 2.2.2.2. That is, DeviceB functions as a translator router because OSPF selects the ABR with the largest router ID as a translator router.

Step 5 # Configure DeviceA as an LSA translator.

```
[~DeviceA] ospf
[*DeviceA-ospf-1] area 1
[*DeviceA-ospf-1-area-0.0.0.1] nssa default-route-advertise no-summary translator-always
[*DeviceA-ospf-1-area-0.0.0.1] commit
[~DeviceA-ospf-1-area-0.0.0.1] quit
```

Step 6 Verify the configuration.

View the OSPF routing table on DeviceC.

```
[~DeviceC] display ospf routing
OSPF Process 1 with Router ID 3.3.3.3
      Routing Tables

      Routing for Network
      Destination Cost Type NextHop AdvRouter Area
      192.168.3.0/24 2 Inter-area 192.168.0.1 1.1.1.1 0.0.0.0
      192.168.4.0/24 2 Inter-area 192.168.2.1 2.2.2.2 0.0.0.0
      192.168.0.0/24 1 Stub 192.168.0.2 3.3.3.3 0.0.0.0
      192.168.1.0/24 2 Inter-area 192.168.2.1 2.2.2.2 0.0.0.0
      192.168.1.0/24 2 Inter-area 192.168.0.1 1.1.1.1 0.0.0.0
      192.168.2.0/24 1 Stub 192.168.2.2 3.3.3.3 0.0.0.0

      Routing for ASEs
      Destination Cost Type Tag NextHop AdvRouter
      10.0.0/8 1 Type2 1 192.168.0.1 1.1.1.1

Total Nets: 7
Intra Area: 2 Inter Area: 4 ASE: 1 NSSA: 0
```

The command output shows that DeviceC has imported an AS external route and that the router ID of the router that advertised the imported AS external route is 1.1.1.1. DeviceA functions as the translator router.

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
#
```

```
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.3.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
ospf 1
area 0.0.0
network 192.168.0.0 0.0.0.255
area 0.0.0.1
network 192.168.1.0 0.0.0.255
network 192.168.3.0 0.0.0.255
nssa default-route-advertise no-summary translator-always
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
router id 2.2.2.2
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.4.2 255.255.255.0
#
ospf 1
area 0.0.0
network 192.168.2.0 0.0.0.255
area 0.0.0.1
network 192.168.1.0 0.0.0.255
network 192.168.4.0 0.0.0.255
nssa
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0
#
ospf 1
area 0.0.0
network 192.168.0.0 0.0.0.255
network 192.168.2.0 0.0.0.255
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
```

```
#  
router id 4.4.4.4  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.3.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.4.1 255.255.255.0  
#  
ospf 1  
import-route static  
area 0.0.0.1  
network 192.168.3.0 0.0.0.255  
network 192.168.4.0 0.0.0.255  
nssa  
#  
ip route-static 10.0.0.0 255.0.0.0 NULL0  
#  
return
```

Example for Configuring OSPF DR Election

This section describes how to set the DR priority on an interface for DR election on a broadcast network.

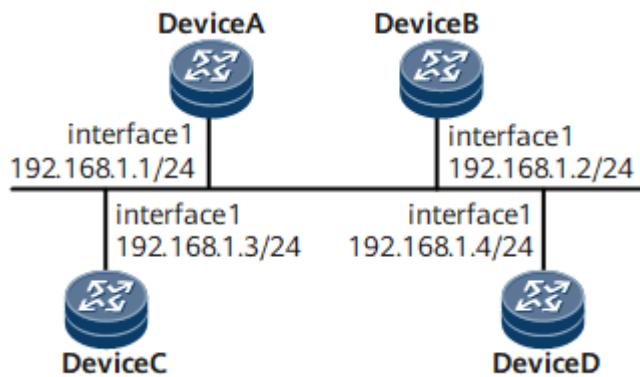
Networking Requirements

On the network shown in [Figure 1-130](#), the interface of DeviceA has the highest priority of 100 on the network and is elected as the DR; DeviceC has the second highest priority and is elected as the BDR. DeviceB has the priority of 0 and cannot be elected as a DR or a BDR; no priority is configured for DeviceD and therefore, DeviceD uses the default value (1).

Figure 1-130 Networking for configuring OSPF DR election



Interface 1 in this example represents GE 1/0/0.



Precautions

Reconfiguring the DR priority for a router does not change the DR or BDR on a network. You can use either of the following methods to re-elect a DR or BDR.

However, the following methods will disconnect OSPF neighbors. Therefore, use the following methods only when necessary.

- Restart the OSPF processes on all routers.
- Configure the **shutdown** and **undo shutdown** commands on the interfaces where the OSPF neighboring relationships are established.
- To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on each router for interconnection.
2. Configure a router ID for each router.
3. Check the DR or BDR status on each router when the default priority is used.
4. Configure the DR priority on an interface and check whether the device is the DR or BDR.

Data Preparation

To complete the configuration, you need the following data:

- Router ID (1.1.1.1) and priority (100) of DeviceA
- Router ID (2.2.2.2) and priority (0) of DeviceB
- Router ID (3.3.3.3) and priority (2) of DeviceC
- Router ID (4.4.4.4) and priority (1) of DeviceD

Procedure

Step 1 Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic OSPF functions.

Configure DeviceA.

```
[~DeviceA] router id 1.1.1.1
[*DeviceA] ospf 1
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0.0] quit
```

Configure DeviceB.

```
[~DeviceB] router id 2.2.2.2
[*DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
```

Configure DeviceC.

```
[~DeviceC] router id 3.3.3.3
[*DeviceC] ospf 1
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
```

Configure DeviceD.

```
[~DeviceD] router id 4.4.4.4
[*DeviceD] ospf 1
[*DeviceD-ospf-1] area 0
[*DeviceD-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0.0] commit
[~DeviceD-ospf-1-area-0.0.0.0] quit
```

Display the status of the DR or BDR.

```
[~DeviceA] display ospf peer
OSPF Process 1 with Router ID 1.1.1.1
Neighbors

Area 0.0.0.0 interface 192.168.1.1 ( GE1/0/0 )'s neighbors
Router ID: 2.2.2.2      Address: 192.168.1.2
  State: 2-Way Mode:Nbr is Slave Priority: 1
  DR: 192.168.1.4  BDR: 192.168.1.3  MTU: 0
  Dead timer due in 32 sec
  Retrans timer interval: 0
  Neighbor is up for 1h15m4s
  Neighbor up time : 2020-06-08 01:41:57
  Authentication Sequence: [ 0 ]

Area 0.0.0.0 interface 192.168.1.1 ( GE1/0/0 )'s neighbors
Router ID: 3.3.3.3      Address: 192.168.1.3
  State: Full Mode:Nbr is Master Priority: 1
  DR: 192.168.1.4  BDR: 192.168.1.3  MTU: 0
  Dead timer due in 34 sec
  Retrans timer interval: 5
  Neighbor is up for 1h15m4s
  Neighbor up time : 2020-06-08 01:41:57
  Authentication Sequence: [ 0 ]

Area 0.0.0.0 interface 192.168.1.1 ( GE1/0/0 )'s neighbors
Router ID: 4.4.4.4      Address: 192.168.1.4
  State: Full Mode:Nbr is Master Priority: 1
  DR: 192.168.1.4  BDR: 192.168.1.3  MTU: 0
  Dead timer due in 32 sec
  Retrans timer interval: 5
  Neighbor is up for 1h15m4s
  Neighbor up time : 2020-06-08 01:41:57
  Authentication Sequence: [ 0 ]
```

Check the neighbor information of DeviceA. You can view the DR priority and the neighbor status. By default, the DR priority is 1. Now DeviceD functions as the DR and DeviceC functions as the BDR.

Step 3 Set the DR priority on each interface.

Configure DeviceA.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ospf dr-priority 100
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

Configure DeviceB.

```
[~DeviceB] interface gigabitethernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] ospf dr-priority 0
```

```
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit

# Configure DeviceC.

[~DeviceC] interface gigabitethernet 1/0/0
[~DeviceC-GigabitEthernet1/0/0] ospf dr-priority 2
[*DeviceC-GigabitEthernet1/0/0] commit
[~DeviceC-GigabitEthernet1/0/0] quit

# Display the status of the DR or BDR.

[~DeviceD] display ospf peer
    OSPF Process 1 with Router ID 4.4.4.4
        Neighbors

    Area 0.0.0.0 interface 192.168.1.4 ( GE1/0/0 )'s neighbors
    Router ID: 1.1.1.1      Address: 192.168.1.1
    State: Full Mode:Nbr is Slave Priority: 100
    DR: 192.168.1.4     BDR: 192.168.1.3     MTU: 0
    Dead timer due in 38 sec
    Retrans timer interval: 5
    Neighbor is up for 2h15m4s
    Neighbor up time : 2020-06-08 01:41:57
    Authentication Sequence: [ 0 ]

    Area 0.0.0.0 interface 192.168.1.4 ( GE1/0/0 )'s neighbors
    Router ID: 2.2.2.2      Address: 192.168.1.2
    State: Full Mode:Nbr is Slave Priority: 0
    DR: 192.168.1.4     BDR: 192.168.1.3     MTU: 0
    Dead timer due in 38 sec
    Retrans timer interval: 5
    Neighbor is up for 2h15m4s
    Neighbor up time : 2020-06-08 01:41:57
    Authentication Sequence: [ 0 ]

    Area 0.0.0.0 interface 192.168.1.4 ( GE1/0/0 )'s neighbors
    Router ID: 3.3.3.3      Address: 192.168.1.3
    State: Full Mode:Nbr is Slave Priority: 2
    DR: 192.168.1.4     BDR: 192.168.1.3     MTU: 0
    Dead timer due in 30 sec
    Retrans timer interval: 5
    Neighbor is up for 2h15m4s
    Neighbor up time : 2020-06-08 01:41:57
    Authentication Sequence: [ 0 ]
```

Step 4 Restart the OSPF processes.

In the user view of each router, run the **reset ospf 1 process** command to restart the OSPF process.

Step 5 Verify the configuration.

Display the status of OSPF neighbors.

```
[~DeviceD] display ospf peer
    OSPF Process 1 with Router ID 4.4.4.4
        Neighbors

    Area 0.0.0.0 interface 192.168.1.4 ( GE1/0/0 )'s neighbors
    Router ID: 1.1.1.1      Address: 192.168.1.1
    State: Full Mode:Nbr is Slave Priority: 100
    DR: 192.168.1.1     BDR: 192.168.1.3     MTU: 0
    Dead timer due in 35 sec
    Retrans timer interval: 5
    Neighbor is up for 3h15m4s
    Neighbor up time : 2020-06-08 01:41:57
    Authentication Sequence: [ 0 ]
```

```
Area 0.0.0 interface 192.168.1.4 ( GE1/0/0 )'s neighbors
Router ID: 2.2.2.2      Address: 192.168.1.2
State: 2-Way Mode:Nbr is Slave Priority: 0
DR: 192.168.1.1 BDR: 192.168.1.3 MTU: 0
Dead timer due in 35 sec
Retrans timer interval: 0
Neighbor is up for 3h15m4s
Neighbor up time : 2020-06-08 01:41:57
Authentication Sequence: [ 0 ]


Area 0.0.0 interface 192.168.1.4 ( GE1/0/0 )'s neighbors
Router ID: 3.3.3.3      Address: 192.168.1.3
State: Full Mode:Nbr is Slave Priority: 2
DR: 192.168.1.1 BDR: 192.168.1.3 MTU: 0
Dead timer due in 37 sec
Retrans timer interval: 5
Neighbor is up for 3h15m4s
Neighbor up time : 2020-06-08 01:41:57
Authentication Sequence: [ 0 ]
```

Display the status of OSPF interfaces.

```
[~DeviceA] display ospf interface
2020-11-21 15:55:20.606

(M) Indicates MADJ interface
      OSPF Process 1 with Router ID 1.1.1.1
      Interfaces

Area: 0.0.0.0 (MPLS TE not enabled)
Interface      IP Address      Type      State  Cost  Pri
GigabitEthernet1/0/0  192.168.1.1  Broadcast  DR    1    100

[~DeviceB] display ospf interface
2020-11-21 15:55:20.606

(M) Indicates MADJ interface

      OSPF Process 1 with Router ID 2.2.2.2
      Interfaces

Area: 0.0.0.0 (MPLS TE not enabled)
Interface      IP Address      Type      State  Cost  Pri
GigabitEthernet1/0/0  192.168.1.2  Broadcast  DROther  1    100
```

If the neighbor is in the Full state, the device has established a neighbor relationship with its neighbor. If the neighbor remains in the 2-Way state, neither of them is the DR or BDR. In this case, they do not need to exchange LSAs.

If the status of the OSPF interface is DROther, the interface is neither DR nor BDR.

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
ospf dr-priority 100
#
ospf 1
area 0.0.0.0
network 192.168.1.0 0.0.0.255
```

- ```

return
```
- DeviceB configuration file

```

sysname DeviceB

router id 2.2.2.2

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
ospf dr-priority 0

ospf 1
area 0.0.0.0
network 192.168.1.0 0.0.0.255

return
```
  - DeviceC configuration file

```

sysname DeviceC

router id 3.3.3.3

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.3 255.255.255.0
ospf dr-priority 2

ospf 1
area 0.0.0.0
network 192.168.1.0 0.0.0.255

return
```
  - DeviceD configuration file

```

sysname DeviceD

router id 4.4.4.4

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.4 255.255.255.0

ospf 1
area 0.0.0.0
network 192.168.1.0 0.0.0.255

return
```

## Example for Configuring OSPF Load Balancing

This section describes how to configure OSPF load balancing, including enabling load balancing and setting priorities for equal-cost routes.

## Networking Requirements

On the network shown in [Figure 1-131](#):

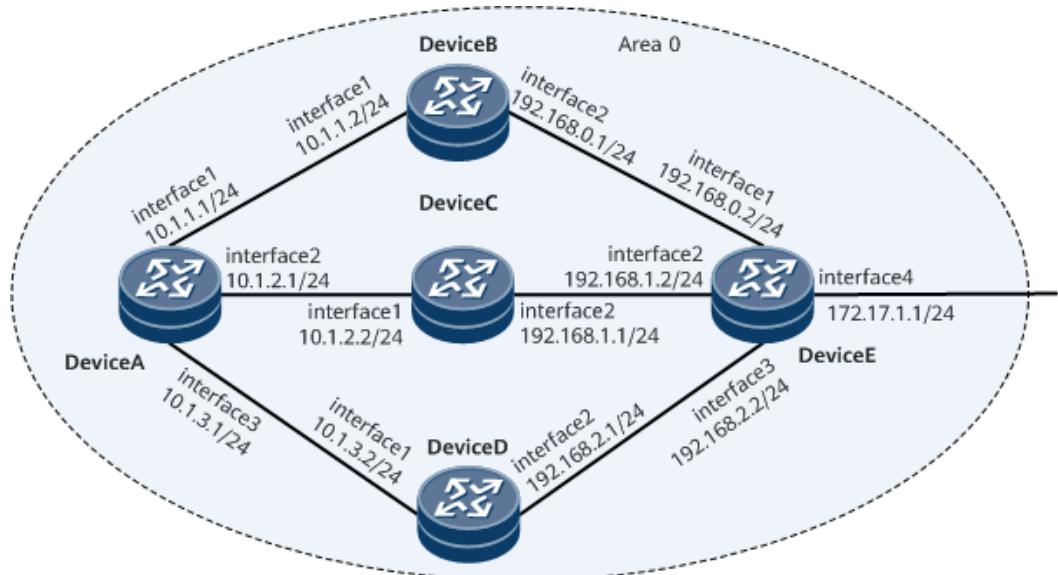
- DeviceA, DeviceB, DeviceC, DeviceD, and DeviceE are connected to each other through OSPF.
- DeviceA, DeviceB, DeviceC, DeviceD, and DeviceE belong to Area 0.

- Load balancing is configured so that the traffic from DeviceA to DeviceE is load-balanced by DeviceC and DeviceD.

**Figure 1-131** Networking for configuring OSPF load balancing

 **NOTE**

In this example, interface1, interface2, interface3, and interface4 represent GE1/0/0, GE2/0/0, GE3/0/0, and GE1/0/1, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

- Configure basic OSPF functions on each router for interconnection.
- Configure load balancing on DeviceA.
- Set the priority for equal-cost routes on DeviceA.
- Configure per-packet load balancing on DeviceA.

## Data Preparation

To complete the configuration, you need the following data.

- Data of DeviceA, including router ID (1.1.1.1), OSPF process ID (1), and network segment addresses of Area 0 (10.1.1.0/24, 10.1.2.0/24 and 10.1.3.0/24)
- Data of DeviceB, including router ID (2.2.2.2), OSPF process ID (1), and network segment addresses of area 0 (10.1.1.0/24 and 192.168.0.0/24)

- Data of DeviceC, including router ID (3.3.3.3), OSPF process ID (1), and network segment addresses of area 0 (10.1.2.0/24 and 192.168.1.0/24)
- Data of DeviceD, including router ID (4.4.4.4), OSPF process ID (1), and network segment addresses of area 0 (10.1.3.0/24 and 192.168.2.0/24)
- Data of DeviceE, including router ID (5.5.5.5), OSPF process ID (1), and network segment addresses of area 0 (192.168.0.0/24, 192.168.1.0/24, 192.168.2.0/24, and 172.17.1.0/24)
- Number of equal-cost routes for load balancing on DeviceA (2)
- Next hop weights of the routes from DeviceA to DeviceB, DeviceC, and DeviceD (2, 1, and 1, respectively)

## Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic OSPF functions. For details, see [Example for Configuring Basic OSPF Functions](#).
- Step 3** Display the routing table of DeviceA.

DeviceA has three valid next hops: DeviceB (10.1.1.2), DeviceC (10.1.2.2), and DeviceD (10.1.3.2) because the default maximum number of equal-cost routes is 64.

```
<DeviceA> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 15 Routes : 15
Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.0/24 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
10.1.1.2/32 Direct 0 0 D 10.1.1.2 GigabitEthernet1/0/0
10.1.2.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet2/0/0
10.1.2.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
10.1.2.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet2/0/0
10.1.3.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet3/0/0
10.1.3.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet3/0/0
10.1.3.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet3/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
192.168.0.0/24 OSPF 10 2 D 10.1.1.2 GigabitEthernet1/0/0
192.168.1.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet2/0/0
192.168.2.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet3/0/0
172.17.1.0/24 OSPF 10 3 D 10.1.1.2 GigabitEthernet1/0/0
 OSPF 10 3 D 10.1.2.2 GigabitEthernet2/0/0
 OSPF 10 3 D 10.1.3.2 GigabitEthernet3/0/0
```

- Step 4** Set the maximum number of routes for load balancing to 2 on DeviceA.

```
[~DeviceA] ospf 1
[*DeviceA-ospf-1] maximum load-balancing 2
[*DeviceA-ospf-1] commit
[~DeviceA-ospf-1] quit
```

# Check the routing table of DeviceA. The command output shows that DeviceA has two routes for load balancing. The maximum number of equal-cost routes is 2. Therefore, the next hops 10.1.1.2 (DeviceB) and 10.1.2.2 (DeviceC) are valid routes.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 15 Routes : 15

Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.0/24 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
10.1.1.2/32 Direct 0 0 D 10.1.1.2 GigabitEthernet1/0/0
10.1.2.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet2/0/0
10.1.2.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
10.1.2.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet2/0/0
10.1.3.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet3/0/0
10.1.3.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet3/0/0
10.1.3.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet3/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
192.168.0.0/24 OSPF 10 2 D 10.1.1.2 GigabitEthernet1/0/0
192.168.1.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet2/0/0
192.168.2.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet3/0/0
172.17.1.0/24 OSPF 10 3 D 10.1.1.2 GigabitEthernet1/0/0
OSPF 10 3 D 10.1.2.2 GigabitEthernet2/0/0
```

### Step 5 Set the priority for equal-cost routes on DeviceA.

```
[~DeviceA] ospf 1
[*DeviceA-ospf-1] nexthop 10.1.1.2 weight 2
[*DeviceA-ospf-1] nexthop 10.1.2.2 weight 1
[*DeviceA-ospf-1] nexthop 10.1.3.2 weight 1
[*DeviceA-ospf-1] commit
[*DeviceA-ospf-1] quit
```

### Step 6 Verify the configuration.

# Check the routing table of DeviceA.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 15 Routes : 15

Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.0/24 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
10.1.1.2/32 Direct 0 0 D 10.1.1.2 GigabitEthernet1/0/0
10.1.2.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet2/0/0
10.1.2.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
10.1.2.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet2/0/0
10.1.3.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet3/0/0
10.1.3.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet3/0/0
10.1.3.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet3/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
192.168.0.0/24 OSPF 10 2 D 10.1.1.2 GigabitEthernet1/0/0
192.168.1.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet2/0/0
192.168.2.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet3/0/0
172.17.1.0/24 OSPF 10 3 D 10.1.2.2 GigabitEthernet2/0/0
OSPF 10 3 D 10.1.3.2 GigabitEthernet3/0/0
```

As shown in the routing table, the priority of the route with 10.1.2.2 and 10.1.3.2 as the next hop addresses is higher than that of the route with 10.1.1.2 as the next hop address. Therefore, DeviceA has only two valid next hops, DeviceC (10.1.2.2) and DeviceD (10.1.3.2).

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0

ospf 1 router-id 1.1.1.1
maximum load-balancing 2
nexthop 10.1.1.2 weight 2
nexthop 10.1.2.2 weight 1
nexthop 10.1.3.2 weight 1
area 0.0.0.
network 10.1.1.0 0.0.0.255
network 10.1.2.0 0.0.0.255
network 10.1.3.0 0.0.0.255

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0

ospf 1 router-id 2.2.2.2
area 0.0.0.
network 10.1.1.0 0.0.0.255
network 192.168.0.0 0.0.255.255

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
ip address 10.1.2.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0

ospf 1 router-id 3.3.3.3
area 0.0.0.
network 10.1.2.0 0.0.0.255
network 192.168.1.0 0.0.0.255

return
```

- DeviceD configuration file

```
#
```

```
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.1 255.255.255.0
#
ospf 1 router-id 4.4.4.4
area 0.0.0
network 10.1.3.0 0.0.0.255
network 192.168.2.0 0.0.0.255
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 172.17.1.1 255.255.255.0
#
ospf 1 router-id 5.5.5.5
area 0.0.0
network 192.168.0.0 0.0.255.255
network 192.168.1.0 0.0.0.255
network 192.168.2.0 0.0.0.255
network 172.17.1.0 0.0.0.255
#
return
```

## Example for Configuring OSPF Fast Convergence

This section describes how to configure OSPF fast convergence by adjusting the timer parameter and configuring BFD.

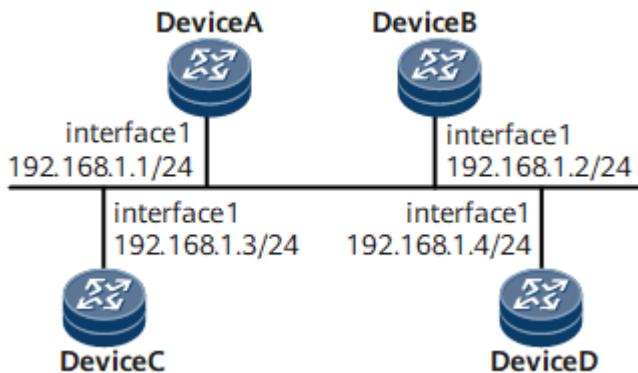
## Networking Requirements

On the broadcast network shown in [Figure 1-132](#), OSPF runs on the four devices, which belong to the same OSPF area.

**Figure 1-132** Networking for configuring OSPF fast convergence



Interface 1 in this example represents GE1/0/0.



## Precautions

When configuring OSPF fast convergence, note the following rules:

- You can decrease the interval at which Hello packets are sent and values of the Dead, Poll, and Wait timers for fast OSPF network convergence. Frequent packet transmission, however, may overload the device. In addition, the OSPF network convergence slows down if the values of these timers are too large. Therefore, set values based on the network stability.
- The intervals at which Hello packets are sent and values of the Dead, Poll, and Wait timers at both ends must be the same. Otherwise, the OSPF neighbor relationship cannot be established.
- To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on each router for interconnection.
2. Configure the BFD function on each router.
3. Adjust the holddown time of the OSPF neighbors on each router.
4. Configure the Smart-discover function on each router.
5. Adjust the intervals for configuration update, LSA reception, and SPF calculation through an intelligent timer on each router.

## Data Preparation

To complete the configuration, you need the following data:

- Holddown time of the OSPF neighbors
- Intervals for LSA update, LSA reception, and SPF calculation

## Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic OSPF functions. For details, see [Example for Configuring Basic OSPF Functions](#).
- Step 3** Adjust the holddown time of the OSPF neighbors on each router.

# Configure Device A.

```
[~DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ospf timer dead 30
[*DeviceA-GigabitEthernet1/0/0] commit
```

### NOTE

- The holddown time of neighbors of the OSPF-capable interfaces must be longer than the interval at which Hello packets are sent, and the values of **dead interval** on the routers in the same network segment must be the same.
- In this configuration example, the following configurations on each router are the same as that on DeviceA. For configuration details, see Configuration Files in this section.
  - Adjust the holddown time of the OSPF neighbors on each router.
  - Configure the Smart-discover function on each router.
  - Adjust the intervals for configuration update, LSA reception, and SPF calculation through an intelligent timer on the router.

- Step 4** Configure the Smart-discover function on each router.

# Configure DeviceA.

```
[*DeviceA-GigabitEthernet1/0/0] ospf smart-discover
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

- Step 5** Configure the BFD function on each router.

# Configure DeviceA.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[~DeviceA] ospf
[*DeviceA-ospf-1] bfd all-interfaces enable
[*DeviceA-ospf-1] commit
[~DeviceA-ospf-1] quit
```

- Step 6** Adjust the intervals for configuration update, LSA reception, and SPF calculation through an intelligent timer on each router.

# Configure DeviceA.

```
[~DeviceA] ospf 1
[~DeviceA-ospf-1] lsa-arrival-interval intelligent-timer 3000 200 500
[*DeviceA-ospf-1] lsa-originate-interval intelligent-timer 3000 200 500
[*DeviceA-ospf-1] spf-schedule-interval intelligent-timer 3000 200 500
[*DeviceA-ospf-1] commit
```

- Step 7** Verify the configuration.

```
Run the display ospf brief command on each router to view the brief information about OSPF. Use Router A as an example. You can view the values of timers.
```

```
[~DeviceA] display ospf brief
 OSPF Process 1 with Router ID 9.9.9.9
 OSPF Protocol Information
 RouterID: 9.9.9.9 Border Router: AREA
 Multi-VPN-Instance is not enabled
 Global DS-TE Mode: Non-Standard IETF Mode
 Graceful-restart capability: disabled
 Helper support capability : not configured
 OSPF Stub Router State Reason: Startup Synchronize
 Router LSA stub links with cost 65535
 Summary LSA with cost 16777214
 External LSA with cost 16777214
 Applications Supported: MPLS Traffic-Engineering
 Spf-schedule-interval: max 10000ms, start 500ms, hold 1000ms
 Default ASE parameters: Metric: 1 Tag: 1 Type: 2
 Route Preference: 10
 ASE Route Preference: 150
 Intra Route Preference: 50
 Inter Route Preference: 50
 SPF Computation Count: 56
 RFC 1583 Compatible
 Retransmission limitation is disabled
 bfd enabled
 BFD Timers: Tx-Interval 10 , Rx-Interval 10 , Multiplier 3
 Area Count: 2 Nssa Area Count: 0
 ExChange/Loading Neighbors: 0

 Area: 0.0.0.0 MPLS TE not enabled
 Authtype: None Area flag: Normal
 SPF scheduled count: 2
 Exchange/Loading neighbors: 0
 Router ID conflict state: Normal

 Interface: 1.1.1.1 (GE3/0/1)
 Cost: 1 State: DR Type: Broadcast MTU: 1500
 Priority: 1
 Designated Router: 1.1.1.1
 Backup Designated Router: 0.0.0.0
 Timers: Hello 10, Dead 40, Wait 40, Poll 120, Retransmit 5, Transmit Delay 1
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

router id 1.1.1.1

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
ospf timer dead 30
ospf smart-discover

ospf 1
bfd all-interfaces enable
spf-schedule-interval intelligent-timer 3000 200 500
lsa-arrival-interval intelligent-timer 3000 200 500
lsa-originate-interval intelligent-timer 3000 200 500
area 0.0.0.0
network 192.168.1.0 0.0.0.255

return
```

- DeviceB configuration file

```

sysname DeviceB

router id 2.2.2.2

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
ospf timer dead 30
ospf smart-discover

ospf 1
bfd all-interfaces enable
spf-schedule-interval intelligent-timer 3000 200 500
lsa-arrival-interval intelligent-timer 3000 200 500
lsa-originate-interval intelligent-timer 3000 200 500
area 0.0.0.0
network 192.168.1.0 0.0.0.255

return
```

- DeviceC configuration file

```

sysname DeviceC

router id 3.3.3.3

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.3 255.255.255.0
ospf timer dead 30
ospf smart-discover

ospf 1
bfd all-interfaces enable
spf-schedule-interval intelligent-timer 3000 200 500
lsa-arrival-interval intelligent-timer 3000 200 500
lsa-originate-interval intelligent-timer 3000 200 500
area 0.0.0.0
network 192.168.1.0 0.0.0.255

return
```

- DeviceD configuration file

```

sysname DeviceD

router id 4.4.4.4

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.4 255.255.255.0
ospf timer dead 30
ospf smart-discover

ospf 1
bfd all-interfaces enable
spf-schedule-interval intelligent-timer 3000 200 500
lsa-arrival-interval intelligent-timer 3000 200 500
lsa-originate-interval intelligent-timer 3000 200 500
area 0.0.0.0
network 192.168.1.0 0.0.0.255
```

```

return
```

## Example for Configuring OSPF Local MT

This section provides an example for configuring OSPF local multicast topology (MT).

### Networking Requirements

When both multicast and MPLS TE tunnels are configured on a network and the TE tunnels are configured with IGP Shortcut, the outbound interface that an IGP calculates for a route may be not a physical interface but a TE tunnel interface. The TE tunnel interface on the Device sends multicast Join packets over a unicast route to the multicast source address. The multicast Join packets are transparent to the Device through which the TE tunnel passes. As a result, the Device through which the TE tunnel passes cannot generate multicast forwarding entries.

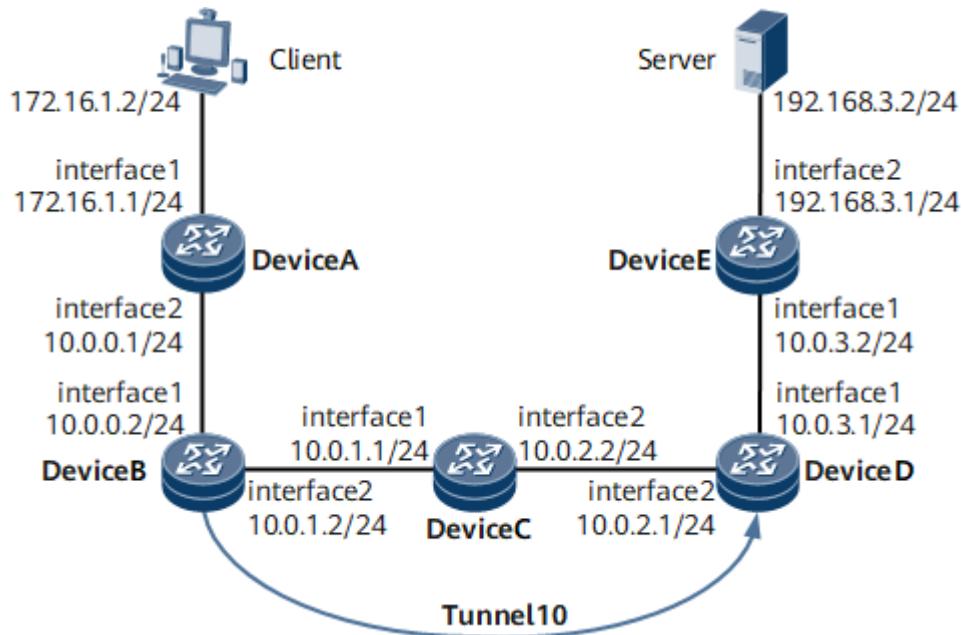
On the network shown in [Figure 1-133](#), DeviceA, DeviceB, DeviceC, DeviceD, and DeviceE are running OSPF. DeviceB and DeviceD set up an MPLS TE tunnel with the tunnel interface Tunnel 10, and IGP Shortcut is enabled on Tunnel 10 of DeviceB. The outbound interface calculated by DeviceB may be the TE tunnel interface, not the physical interface GE 2/0/0. Interface TE tunnel 10 on DeviceB sends multicast Join messages over a unicast route to the multicast source address. The multicast Join messages are transparent to DeviceC spanned by TE tunnel 10. As a result, DeviceC cannot generate multicast forwarding entries.

To resolve the problem, enable OSPF local MT on DeviceB. After local MT is enabled, if the outbound interface of a calculated route is an IGP Shortcut-enabled TE tunnel interface, the route management (RM) module creates an independent Multicast IGP (MIGP) routing table for the multicast protocol, calculates a physical outbound interface for the route, and adds the route to the MIGP routing table. Multicast packets are then forwarded along this route.

**Figure 1-133** Configuring local OSPF MT

 NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable OSPF and configure basic OSPF functions on each Device.
2. Enable PIM-SM on each Device.
3. Configure an MPLS Resource Reservation Protocol (RSVP)-TE tunnel.
4. Configure an MPLS TE tunnel and enable IGP Shortcut for it on DeviceB.
5. Enable OSPF local MT on DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

- **Table 1-51** lists the IP addresses of interfaces on the router.

**Table 1-51** IP address of each interface

| Device  | IP Address of Loopback 0 |
|---------|--------------------------|
| DeviceA | 1.1.1.1/32               |
| DeviceB | 2.2.2.2/32               |

| Device  | IP Address of Loopback 0 |
|---------|--------------------------|
| DeviceC | 3.3.3.3/32               |
| DeviceD | 4.4.4.4/32               |
| DeviceE | 5.5.5.5/32               |

- The TE tunnel interface Tunnel 10 uses the IP address of Loopback 0 and runs the MPLS TE protocol. The destination address of the TE tunnel is 4.4.4.4, and the tunnel ID is 100. The TE tunnel uses RSVP-TE as a signaling protocol.

## Procedure

**Step 1** Assign an IP address to each interface.

[Figure 1-133](#) shows how to assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic OSPF functions.

[Configuring Basic OSPF Functions](#) shows how to configure basic OSPF functions. For configuration details, see [Configuration Files](#) in this section.

**Step 3** Configure Protocol Independent Multicast-Sparse Mode (PIM-SM).

# Enable PIM-SM on DeviceA.

```
[~DeviceA] multicast routing-enable
[*DeviceA] interface Gigabitethernet 2/0/0
[~DeviceA-GigabitEthernet2/0/0] pim sm
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] interface Gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] pim sm
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```



Enable PIM-SM on each Device. The configurations for DeviceB, DeviceC, DeviceD, and DeviceE are similar to those on DeviceA. For configuration details, see [Configuration Files](#) in this section.

# Enable Internet Group Management Protocol (IGMP) on GE 1/0/0 of DeviceA.

```
[~DeviceA] interface Gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] igmp enable
[*DeviceA-GigabitEthernet1/0/0] igmp version 3
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure a C-BSR and a C-RP. Set the service range of the RP on DeviceD and specify the locations of the C-BSR and the C-RP.

```
[~DeviceD] pim
[*DeviceD-pim] c-bsr LoopBack0
[*DeviceD-pim] c-rp LoopBack0
[*DeviceD-pim] commit
[~DeviceD-pim] quit
```

# Check the multicast routing table on DeviceC.

```
[~DeviceC] display multicast routing-table
```

```
Multicast routing table of VPN-Instance: public net
Total 3 entries

00001. (192.168.3.8, 224.31.31.31)
 Uptime: 15:03:04
 Upstream Interface: GigabitEthernet2/0/0
 List of 1 downstream interface
 1: GigabitEthernet1/0/0

00002. (192.168.3.9, 224.31.31.31)
 Uptime: 15:03:04
 Upstream Interface: GigabitEthernet2/0/0
 List of 1 downstream interface
 1: GigabitEthernet1/0/0

00003. (192.168.3.10, 224.31.31.31)
 Uptime: 15:03:04
 Upstream Interface: GigabitEthernet2/0/0
 List of 1 downstream interface
 1: GigabitEthernet1/0/0
```

The preceding command output shows information about the multicast routing table on DeviceC.

**Step 4** Configure an MPLS RSVP-TE tunnel.

# Configure DeviceB.

```
[~DeviceB] mpls lsr-id 2.2.2.2
[*DeviceB] mpls
[*DeviceB-mpls] mpls te
[*DeviceB-mpls] mpls rsvp-te
[*DeviceB-mpls] mpls te cspf
[*DeviceB-mpls] commit
[*DeviceB-mpls] quit
[*DeviceB] interface Gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] mpls
[*DeviceB-GigabitEthernet2/0/0] mpls te
[*DeviceB-GigabitEthernet2/0/0] mpls rsvp-te
[*DeviceB-GigabitEthernet2/0/0] commit
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] ospf 1
[*DeviceB-ospf-1] enable traffic-adjustment
[*DeviceB-ospf-1] opaque-capability enable
[*DeviceB-ospf-1] area 0.0.0.0
[*DeviceB-ospf-1-area-0.0.0.0] mpls-te enable
[*DeviceB-ospf-1-area-0.0.0.0] commit
[*DeviceB-ospf-1-area-0.0.0.0] quit
```

# Configure DeviceC.

```
[~DeviceC] mpls lsr-id 3.3.3.3
[*DeviceC] mpls
[*DeviceC-mpls] mpls te
[*DeviceC-mpls] mpls rsvp-te
[*DeviceC-mpls] commit
[*DeviceC-mpls] quit
[*DeviceC] interface Gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] mpls
[*DeviceC-GigabitEthernet1/0/0] mpls te
[*DeviceC-GigabitEthernet1/0/0] mpls rsvp-te
[*DeviceC-GigabitEthernet1/0/0] commit
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface Gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] mpls
[*DeviceC-GigabitEthernet2/0/0] mpls te
[*DeviceC-GigabitEthernet2/0/0] mpls rsvp-te
[*DeviceC-GigabitEthernet2/0/0] commit
[*DeviceC-GigabitEthernet2/0/0] quit
```

```
[*DeviceC] ospf 1
[*DeviceC-ospf-1] opaque-capability enable
[*DeviceC-ospf-1] area 0.0.0.0
[*DeviceC-ospf-1-area-0.0.0.0] mpls-te enable
[*DeviceC-ospf-1-area-0.0.0.0] commit
[*DeviceC-ospf-1-area-0.0.0.0] quit
```

# Configure DeviceD.

```
[~DeviceD] mpls lsr-id 4.4.4.4
[~DeviceD] mpls
[*DeviceD-mpls] mpls te
[*DeviceD-mpls] mpls rsvp-te
[*DeviceD-mpls] commit
[*DeviceD-mpls] quit
[~DeviceD] interface GigabitEthernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] mpls
[*DeviceD-GigabitEthernet2/0/0] mpls te
[*DeviceD-GigabitEthernet2/0/0] mpls rsvp-te
[*DeviceD-GigabitEthernet2/0/0] commit
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] ospf 1
[*DeviceD-ospf-1] opaque-capability enable
[*DeviceD-ospf-1] area 0.0.0.0
[*DeviceD-ospf-1-area-0.0.0.0] mpls-te enable
[*DeviceD-ospf-1-area-0.0.0.0] commit
[*DeviceD-ospf-1-area-0.0.0.0] quit
```

### Step 5 Configure an MPLS TE tunnel and enable IGP Shortcut.

# Configure an MPLS TE tunnel and enable IGP Shortcut for it on DeviceB.

```
[*DeviceB] interface Tunnel 10
[*DeviceB-Tunnel10] ip address unnumbered interface loopback 0
[*DeviceB-Tunnel10] tunnel-protocol mpls te
[*DeviceB-Tunnel10] destination 4.4.4.4
[*DeviceB-Tunnel10] mpls te tunnel-id 100
[*DeviceB-Tunnel10] mpls te igp shortcut ospf
[*DeviceB-Tunnel10] mpls te igp metric relative -10
[*DeviceB-Tunnel10] commit
[*DeviceB-Tunnel10] quit
```

# Check the OSPF routing table on DeviceB. Information shows that an MPLS TE tunnel has been set up.

```
[*DeviceB] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 14 Routes : 14

Destination/Mask Proto Pre Cost Flags NextHop Interface
2.2.2.2/32 Direct 0 0 D 127.0.0.1 InLoopBack0
3.3.3.3/32 OSPF 10 1 D 10.0.1.1 GigabitEthernet2/0/0
4.4.4.4/32 OSPF 10 1 D 2.2.2.2 Tunnel10
5.5.5.5/32 OSPF 10 2 D 2.2.2.2 Tunnel10
10.0.0.0/24 Direct 0 0 D 10.0.0.2 GigabitEthernet1/0/0
10.0.0.2/32 Direct 0 0 D 127.0.0.1 InLoopBack0
10.0.1.0/24 Direct 0 0 D 10.0.1.2 GigabitEthernet2/0/0
10.0.1.2/32 Direct 0 0 D 127.0.0.1 InLoopBack0
10.0.2.0/24 OSPF 10 2 D 10.0.1.1 GigabitEthernet2/0/0
 OSPF 10 2 D 10.0.1.1 Tunnel10
10.0.3.0/24 OSPF 10 2 D 2.2.2.2 Tunnel10
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
172.16.1.0/24 OSPF 10 2 D 10.0.0.1 GigabitEthernet2/0/0
192.168.3.0/24 OSPF 10 3 D 2.2.2.2 Tunnel10
```

# Check the multicast routing table on DeviceC.

```
[~DeviceC] display multicast routing-table
```

No multicast entry is displayed, indicating that multicast packets are discarded.

**Step 6** Enable OSPF local MT.

```
Enable OSPF local MT on DeviceB.
```

```
[*DeviceB] ospf
[*DeviceB-ospf-1] local-mt enable
[*DeviceB-ospf-1] commit
[*DeviceB-ospf-1] quit
```

**Step 7** Verify the configuration.

```
Check the multicast routing table on DeviceC.
```

```
[*DeviceC] display multicast routing-table
Multicast routing table of VPN-Instance: public net
Total 3 entries

00001. (192.168.3.8, 224.31.31.31)
 Uptime: 00:00:19
 Upstream Interface: GigabitEthernet2/0/0
 List of 1 downstream interface
 1: GigabitEthernet1/0/0

00002. (192.168.3.9, 224.31.31.31)
 Uptime: 00:00:19
 Upstream Interface: GigabitEthernet2/0/0
 List of 1 downstream interface
 1: GigabitEthernet1/0/0

00003. (192.168.3.10, 224.31.31.31)
 Uptime: 00:00:19
 Upstream Interface: GigabitEthernet2/0/0
 List of 1 downstream interface
 1: GigabitEthernet1/0/0
```

The preceding command output shows information about the multicast routing table on DeviceC.

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
router id 1.1.1.1
#
multicast routing-enable
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.0.1 255.255.255.0
pim sm
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
pim sm
igmp enable
igmp version 3
#
interface LoopBack0
ip address 1.1.1.1 255.255.255.255
```

```

ospf 1
area 0.0.0.
 network 172.16.1.0 0.0.0.255
 network 10.0.0.0 0.0.0.255
 network 1.1.1.1 0.0.0.0

return
```

- DeviceB configuration file

```

sysname DeviceB

router id 2.2.2.2

multicast routing-enable

mpls lsr-id 2.2.2.2

mpls
 mpls te
 mpls rsvp-te
 mpls te cspf

ospf 1
 opaque-capability enable
 enable traffic-adjustment
 local-mt enable
area 0.0.0.
 network 10.0.0.0 0.0.0.255
 network 10.0.1.0 0.0.0.255
 network 2.2.2.2 0.0.0.0
 mpls-te enable

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.0.0.2 255.255.255.0
pim sm

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.1.2 255.255.255.0
pim sm
mpls
 mpls te
 mpls rsvp-te

interface LoopBack0
ip address 2.2.2.2 255.255.255.255
pim sm

interface Tunnel10
ip address unnumbered interface LoopBack0
tunnel-protocol mpls te
destination 4.4.4.4
mpls te tunnel-id 100
mpls te igp shortcut ospf
mpls te igp metric relative -10

pim
 C-BSR LoopBack0
 C-RP LoopBack0

return
```

- DeviceC configuration file

```

sysname DeviceC

router id 3.3.3.3
```

```

multicast routing-enable

mpls lsr-id 3.3.3.3

mpls
mpls te
mpls rsvp-te

ospf 1
opaque-capability enable
area 0.0.0
network 10.0.1.0 0.0.0.255
network 10.0.2.0 0.0.0.255
network 3.3.3.3 0.0.0.0
mpls-te enable

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.0.1.1 255.255.255.0
pim sm
mpls
mpls te
mpls rsvp-te

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.2.2 255.255.255.0
pim sm
mpls
mpls te
mpls rsvp-te

interface LoopBack0
undo shutdown
ip address 3.3.3.3 255.255.255.255

return
```

- DeviceD configuration file

```

sysname DeviceD

router id 4.4.4.4

multicast routing-enable

mpls lsr-id 4.4.4.4

mpls
mpls te
mpls rsvp-te

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.0.3.1 255.255.255.0
pim sm

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.2.1 255.255.255.0
pim sm
mpls
mpls te
mpls rsvp-te

interface LoopBack0
ip address 4.4.4.4 255.255.255.255
pim sm
#
```

```
ospf 1
opaque-capability enable
area 0.0.0
network 10.0.2.0 0.0.0.255
network 10.0.3.0 0.0.0.255
network 4.4.4.4 0.0.0.0
mpls-te enable
#
pim
C-BSR LoopBack0
C-RP LoopBack0
#
return
```

- Device E configuration file

```
#
sysname DeviceE
#
router id 5.5.5.5
#
multicast routing-enable
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.0.3.2 255.255.255.0
pim sm
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.3.1 255.255.255.0
pim sm
#
interface LoopBack0
ip address 5.5.5.5 255.255.255.255
pim sm
#
ospf 1
area 0.0.0.0
network 10.0.3.0 0.0.0.255
network 192.168.3.0 0.0.0.255
network 5.5.5.5 0.0.0.0
#
return
```

## Example for Configuring OSPF IP FRR

This section describes how to configure OSPF IP FRR with an example, including how to block FRR on certain interfaces to prevent the links connected to these interfaces from functioning as backup links and how to bind OSPF IP FRR to a BFD session.

## Networking Requirements

When a fault occurs on the primary link T, traffic is switched to a backup link. In such a scenario, two problems arise:

- It takes hundreds of milliseconds for the traffic to be switched to a backup link. During this period, services are interrupted.
- Traffic may be switched to the link that passes through DeviceA. DeviceA is an ASBR and is not expected to function as a backup device.

If a fault occurs on the network, OSPF IP FRR can fast switch traffic to the backup link without waiting for route convergence. This ensures uninterrupted traffic transmission. In addition, you can also prevent the link which passes through DeviceA from functioning as the FRR backup link.

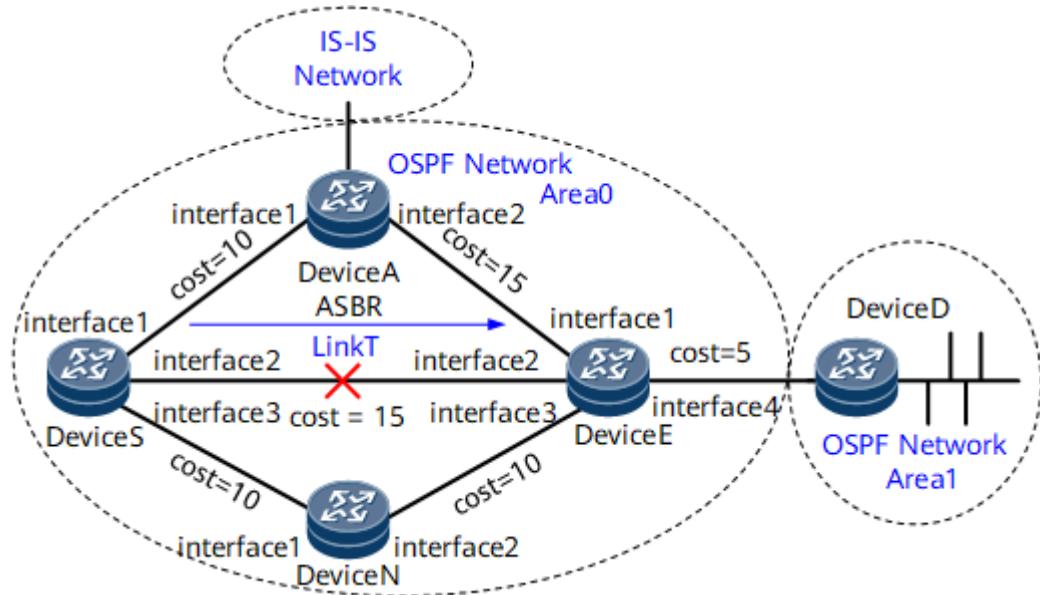
On the network shown in [Figure 1-134](#):

- All routers run OSPF.
- The link cost meets the OSPF IP FRR traffic protection inequality.
- If the primary link T fails, Device S immediately switches traffic to the backup link which passes through DeviceN.
- Based on the network planning, the link which passes through DeviceA does not function as an FRR backup link.

[Figure 1-134](#) Networking for configuring OSPF IP FRR

NOTE

Interfaces 1 through 4 in this example represent GE 1/0/0, GE 2/0/0, GE 3/0/0, and GE 1/0/1, respectively.



## Precautions

During the configuration, pay attention to the following points:

- Before configuring OSPF IP FRR, block FRR on certain interfaces to prevent the links connected to these interfaces from functioning as backup links. After that, the link where the interface resides is not calculated as a backup link during FRR calculation.
- During the configuration of OSPF IP FRR, the lower layer needs to fast respond to a link change so that traffic can be rapidly switched to the backup link. After the **bfd all-interfaces frr-binding** command is run, the BFD session status is bound to the link status of the interface (when the BFD session goes down, the link status of the interface also goes down). In this manner, faults can be rapidly detected.
- To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on each router.
2. Configure BFD for OSPF on all the devices in Area 0.
3. Set the costs of links to ensure that link T is selected to transmit traffic.
4. Block FRR on a specified interface of DeviceS.
5. Enable OSPF IP FRR on DeviceS to protect the traffic forwarded by DeviceS.

## Data Preparation

To complete the configuration, you need the following data:

| Device  | Router ID | Interface | IP Address    |
|---------|-----------|-----------|---------------|
| DeviceS | 1.1.1.1   | GE1/0/0   | 10.1.1.1/24   |
|         |           | GE2/0/0   | 10.1.2.1/24   |
|         |           | GE3/0/0   | 10.1.3.1/24   |
| DeviceA | 2.2.2.2   | GE1/0/0   | 10.1.1.2/24   |
|         |           | GE2/0/0   | 10.2.1.2/24   |
| DeviceN | 3.3.3.3   | GE1/0/0   | 10.1.3.2/24   |
|         |           | GE2/0/0   | 10.2.3.2/24   |
| DeviceE | 4.4.4.4   | GE1/0/0   | 10.2.1.1/24   |
|         |           | GE2/0/0   | 10.1.2.2/24   |
|         |           | GE3/0/0   | 10.2.3.1/24   |
|         |           | GE1/0/1   | 172.17.1.1/24 |

## Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic OSPF functions. For details, see [Example for Configuring Basic OSPF Functions](#).
- Step 3** Configure BFD for OSPF on all the devices in Area 0. For details, see [Example for Configuring BFD for OSPF](#).
- Step 4** Set the costs of links to ensure that link T is selected to transmit traffic.

# Configure DeviceS.

```
[~DeviceS] interface gigabitethernet1/0/0
[*DeviceS-GigabitEthernet1/0/0] ospf cost 10
```

```
[*DeviceS-GigabitEthernet1/0/0] quit
[*DeviceS] interface gigabitethernet2/0/0
[*DeviceS-GigabitEthernet2/0/0] ospf cost 15
[*DeviceS-GigabitEthernet2/0/0] quit
[*DeviceS] interface gigabitethernet3/0/0
[*DeviceS-GigabitEthernet3/0/0] ospf cost 10
[*DeviceS-GigabitEthernet3/0/0] quit
[*DeviceS] commit
```

# Configure DeviceA.

```
[~DeviceA] interface gigabitethernet2/0/0
[*DeviceA-GigabitEthernet2/0/0] ospf cost 15
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

# Configure DeviceN.

```
[~DeviceN] interface gigabitethernet2/0/0
[*DeviceN-GigabitEthernet2/0/0] ospf cost 10
[*DeviceN-GigabitEthernet2/0/0] quit
[*DeviceN] commit
```

**Step 5** Block FRR on a specified interface of DeviceS.

```
[~DeviceS] interface gigabitethernet1/0/0
[*DeviceS-GigabitEthernet1/0/0] ospf frr block
[*DeviceS-GigabitEthernet1/0/0] quit
[*DeviceS] commit
```

**Step 6** Enable OSPF IP FRR on DeviceS.

# Enable OSPF IP FRR on DeviceS.

```
[~DeviceS] ospf
[*DeviceS-ospf-1] frr
[*DeviceS-ospf-1-frr] loop-free-alternate
[*DeviceS-ospf-1-frr] commit
```

**Step 7** Verify the configuration.

# Run the **display ospf routing router-id** command on DeviceS to view routing information.

```
[~DeviceS-ospf-1-frr] display ospf routing router-id 4.4.4.4
OSPF Process 1 with Router ID 1.1.1.1

Destination : 4.4.4.4 Route Type : Intra-area
Area : 0.0.0.1 AdvRouter : 4.4.4.4
Type : ASBR
URT Cost : 59
NextHop : 10.2.2.1. Interface : GE2/0/0
Backup Nexthop : 10.1.3.2 Backup Interface : GE3/0/0 Backup Type : LFA LINK
BakLabelStack : {48092,48092}
```

The preceding display shows that a backup route is generated on DeviceS.

----End

## Configuration Files

- Device S configuration file

```

sysname DeviceS

bfd

interface GigabitEthernet1/0/0
```

```
ip address 10.1.1.1 255.255.255.0
ospf frr block
ospf cost 10
#
interface GigabitEthernet2/0/0
ip address 10.1.2.1 255.255.255.0
ospf cost 15
#
interface GigabitEthernet3/0/0
ip address 10.1.3.1 255.255.255.0
ospf cost 10
#
interface LoopBack0
ip address 1.1.1.1 255.255.255.255
#
ospf 1 router-id 1.1.1.1
bfd all-interfaces enable
bfd all-interfaces frr-binding
frr
loop-free-alternate
area 0.0.0
network 10.1.1.0 0.0.0.255
network 10.1.2.0 0.0.0.255
network 10.1.3.0 0.0.0.255
#
return
```

- DeviceA configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet1/0/0
ip address 10.1.1.2 255.255.255.0
ospf cost 10
#
interface GigabitEthernet2/0/0
ip address 10.2.1.2 255.255.255.0
ospf cost 15
#
interface LoopBack0
ip address 2.2.2.2 255.255.255.255
#
ospf 1 router-id 2.2.2.2
bfd all-interfaces enable
bfd all-interfaces frr-binding
frr
loop-free-alternate
area 0.0.0
network 10.1.1.0 0.0.0.255
network 10.2.2.0 0.0.0.255
#
return
```

- DeviceN configuration file

```
#
sysname DeviceN
#
bfd
#
interface GigabitEthernet1/0/0
ip address 10.1.3.2 255.255.255.0
ospf cost 10
#
interface GigabitEthernet2/0/0
ip address 10.2.3.2 255.255.255.0
ospf cost 10
#
interface LoopBack0
```

```
ip address 3.3.3.3 255.255.255.255
#
ospf 1 router-id 3.3.3.3
bfd all-interfaces enable
bfd all-interfaces frr-binding
frr
area 0.0.0.0
network 10.1.3.0 0.0.0.255
network 10.2.3.0 0.0.0.255
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
bfd
#
interface GigabitEthernet1/0/0
ip address 10.2.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet3/0/0
ip address 10.2.3.1 255.255.255.0
#
interface GigabitEthernet1/0/1
ip address 172.17.1.1 255.255.255.0
ospf cost 5
#
interface LoopBack0
ip address 4.4.4.4 255.255.255.255
#
ospf 1 router-id 4.4.4.4
bfd all-interfaces enable
bfd all-interfaces frr-binding
area 0.0.0.0
network 10.1.1.0 0.0.0.255
network 10.1.2.0 0.0.0.255
network 10.1.3.0 0.0.0.255
network 172.17.1.0 0.0.0.255
#
return
```

## Example for Configuring BFD for OSPF

This section describes how to configure BFD for OSPF. After BFD for OSPF is configured, BFD can fast detect link faults and report them to OSPF so that service traffic can be transmitted through the backup link.

## Networking Requirements

OSPF enables the device to periodically send Hello packets to a neighboring device for fault detection. Detecting a fault takes more than 1s. With the development of technologies, voice, video, and other VOD services are widely used. These services are quite sensitive to packet loss and delays. When traffic is transmitted at gigabit rates, long-time fault detection will cause packet loss. This cannot meet high reliability requirements of the carrier-class network. BFD for OSPF is used to resolve the problem. After BFD for OSPF is configured, the link status can be rapidly detected and fault detection can be completed in milliseconds. This speeds up OSPF convergence when the link status changes.

For example, as shown in [Figure 1-135](#), the primary link DeviceA -> DeviceB and the secondary link DeviceA -> DeviceC -> DeviceB are deployed on the network.

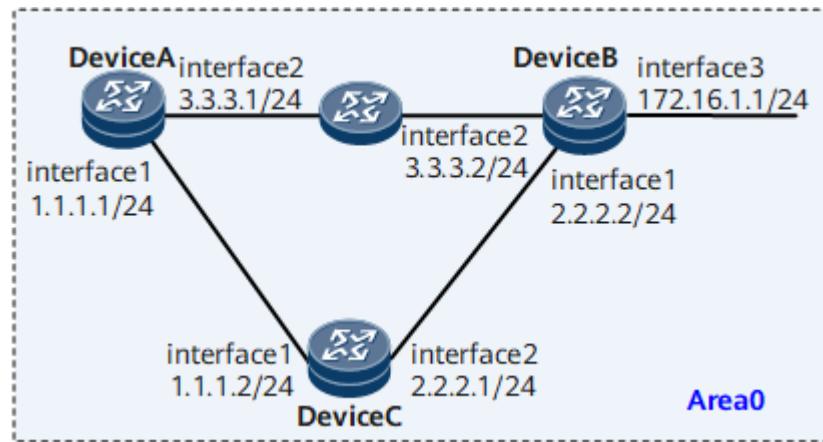
Traffic is transmitted on the primary link normally. When the primary link becomes faulty, the Device is expected to rapidly detect the fault and switch traffic to the secondary link.

You can configure BFD for OSPF to detect the OSPF neighbor relationship between DeviceA and DeviceB. When the link between DeviceA and DeviceB fails, BFD can rapidly detect the failure and report it to OSPF. Traffic is then switched to the secondary link.

**Figure 1-135 Networking for configuring BFD for OSPF**

 **NOTE**

Interfaces 1 through 3 in this example represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on each router for interconnection.
2. Enable global BFD.
3. Enable OSPF BFD on DeviceA and DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

- Data of DeviceA, including the router ID (1.1.1.1), OSPF process number (1), and network segment addresses of Area 0 (3.3.3.0/24 and 1.1.1.0/24)

- Data of DeviceB, including the router ID (2.2.2.2), OSPF process number (1), and network segment addresses of Area 0 (3.3.3.0/24, 2.2.2.0/24, and 172.16.1.0/24)
- Data of DeviceC, including the router ID (3.3.3.3), OSPF process number (1), and network segment addresses of Area 0 (1.1.1.0/24 and 2.2.2.0/24)
- Minimum interval at which BFD packets are received and sent and local detection multiplier on DeviceA and DeviceB

## Procedure

**Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic OSPF functions.

# Configure DeviceA.

```
[~DeviceA] router id 1.1.1.1
[*DeviceA] ospf 1
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0.0] network 1.1.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.0] network 3.3.3.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0.0] quit
[~DeviceA-ospf-1] quit
```

# Configure DeviceB.

```
[~DeviceB] router id 2.2.2.2
[*DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 2.2.2.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 3.3.3.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 172.16.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
[~DeviceB-ospf-1] quit
```

# Configure DeviceC.

```
[~DeviceC] router id 3.3.3.3
[*DeviceC] ospf 1
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 1.1.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] network 2.2.2.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
[~DeviceC-ospf-1] quit
```

# After the preceding configurations, run the **display ospf peer** command. You can view that a neighbor relationship is established between DeviceA and DeviceB, and between DeviceB and DeviceC. The following example uses the command output on DeviceA.

```
<DeviceA> display ospf peer
(M) Indicates MADJ neighbor
 OSPF Process 1 with Router ID 1.1.1.1
 Neighbors

Area 0.0.0.0 interface 3.3.3.1 (GE2/0/0)'s neighbors
Router ID: 2.2.2.2 Address: 3.3.3.2
 State: Full Mode:Nbr is Master Priority: 1
 DR: 3.3.3.1 BDR: 3.3.3.2 MTU: 0
 Dead timer due in 35 sec
```

```
Retrans timer interval: 5
Neighbor is up for 1h15m4s
Neighbor up time : 2020-06-08 01:41:57
Authentication Sequence: [0]

Area 0.0.0.0 interface 1.1.1.1 (GE1/0/0)'s neighbors
Router ID: 3.3.3.3 Address: 1.1.1.2
State: Full Mode:Nbr is Master Priority: 1
DR: 1.1.1.1 BDR: 1.1.1.2 MTU: 0
Dead timer due in 39 sec
Retrans timer interval: 5
Neighbor is up for 1h15m4s
Neighbor up time : 2020-06-08 01:41:57
Authentication Sequence: [0]
```

# Display information about the OSPF routing table on DeviceA, and you can view the routing entries to DeviceB and DeviceC. The next hop address of the route to 172.16.1.0/24 is 3.3.3.2, and service traffic is transmitted on the primary link (DeviceA → DeviceB).

```
<DeviceA> display ospf routing
OSPF Process 1 with Router ID 1.1.1.1
Routing Tables

Routing for Network
Destination Cost Type NextHop AdvRouter Area
2.2.2.0/24 2 Stub 1.1.1.2 3.3.3.3 0.0.0.0
172.16.1.0/24 2 Stub 3.3.3.2 2.2.2.2 0.0.0.0

Total Nets: 2
Intra Area: 2 Inter Area: 0 ASE: 0 NSSA: 0
```

### Step 3 Configure OSPF BFD.

# Enable global BFD on DeviceA.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] ospf 1
[*DeviceA-ospf-1] bfd all-interfaces enable
[*DeviceA-ospf-1] commit
[~DeviceA-ospf-1] quit
```

# Enable global BFD on DeviceB.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] ospf 1
[*DeviceB-ospf-1] bfd all-interfaces enable
[*DeviceB-ospf-1] commit
[~DeviceB-ospf-1] quit
```

# Enable global BFD on DeviceC.

```
[~DeviceC] bfd
[*DeviceC-bfd] quit
[*DeviceC] ospf 1
[*DeviceC-ospf-1] bfd all-interfaces enable
[*DeviceC-ospf-1] commit
[~DeviceC-ospf-1] quit
```

# After the preceding configurations, run the **display ospf bfd session all** command on DeviceA, DeviceB, or DeviceC. You can view that the BFD session is Up.

Use the command output DeviceA as an example.

```
[~DeviceA] display ospf bfd session all
```

```
OSPF Process 1 with Router ID 1.1.1.1
Area 0.0.0 interface 1.1.1.1(GE1/0/0)'s BFD Sessions

NeighborId:2.2.2.2 Areald:0.0.0.0 Interface:GE1/0/0
BFDState:up rx :1000 tx :1000
Multiplier:3 BFD Local Dis:0 LocalIpAdd:1.1.1.1
RemotelpAdd:1.1.1.2 Diagnostic Info:0

Area 0.0.0 interface 3.3.3.1(GE2/0/0)'s BFD Sessions

NeighborId:3.3.3.3 Areald:0.0.0.0 Interface:GE2/0/0
BFDState:up rx :1000 tx :1000
Multiplier:3 BFD Local Dis:0 LocalIpAdd:3.3.3.1
RemotelpAdd:3.3.3.2 Diagnostic Info:0
```

**Step 4** Configure BFD on an interface.

# Configure BFD on GigabitEthernet 2/0/0 of DeviceA. Set the minimum intervals at which packets are received and sent to 500 ms and the local detection multiplier to 4.

```
[~DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] ospf bfd enable
[*DeviceA-GigabitEthernet2/0/0] ospf bfd min-tx-interval 500 min-rx-interval 500 detect-multiplier 4
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

# Configure BFD on GigabitEthernet 2/0/0 of DeviceB. Set the minimum intervals at which packets are received and sent to 500 ms and the local detection multiplier to 4.

```
[~DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] ospf bfd enable
[*DeviceB-GigabitEthernet2/0/0] ospf bfd min-tx-interval 500 min-rx-interval 500 detect-multiplier 4
[*DeviceB-GigabitEthernet2/0/0] commit
[~DeviceB-GigabitEthernet2/0/0] quit
```

# After the preceding configurations, run the **display ospf bfd session all** command on DeviceA or DeviceB. You can check that the minimum intervals at which packets are sent and received are 500 ms and that the local detection multiplier is 4.

Use the command output DeviceB as an example.

```
[~DeviceB] display ospf bfd session all
OSPF Process 1 with Router ID 2.2.2.2
Area 0.0.0 interface 3.3.3.2(GE2/0/0)'s BFD Sessions

NeighborId:1.1.1.1 Areald:0.0.0.0 Interface: GigabitEthernet2/0/0
BFDState:up rx :500 tx :500
Multiplier:4 BFD Local Dis:0 LocalIpAdd:3.3.3.2
RemotelpAdd:3.3.3.1 Diagnostic Info:0

Area 0.0.0 interface 2.2.2.2(GE1/0/0)'s BFD Sessions

NeighborId:3.3.3.3 Areald:0.0.0.0 Interface: GE1/0/0
BFDState:up rx :1000 tx :1000
Multiplier:3 BFD Local Dis:0 LocalIpAdd:2.2.2.2
RemotelpAdd:2.2.2.1 Diagnostic Info:0
```

**Step 5** Verify the configuration.

# Run the **shutdown** command on GigabitEthernet 2/0/0 of DeviceB to simulate a primary link failure.

```
[~DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] shutdown
```

```
[*DeviceB-GigabitEthernet2/0/0] commit
```

```
Display the routing table on routerDeviceA. You can view that the backup link
(DeviceA - DeviceC - DeviceB) takes effect after the primary link fails and that the
next hop address of the route to 172.16.1.0/24 is 1.1.1.2.
```

```
<DeviceA> display ospf routing
OSPF Process 1 with Router ID 1.1.1.1
Routing Tables

Routing for Network
Destination Cost Type NextHop AdvRouter Area
2.2.2.0/24 2 Stub 1.1.1.2 3.3.3.3 0.0.0.0
172.16.1.0/24 3 Stub 1.1.1.2 2.2.2.2 0.0.0.0

Total Nets: 2
Intra Area: 2 Inter Area: 0 ASE: 0 NSSA: 0
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

router id 1.1.1.1

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 1.1.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 3.3.3.1 255.255.255.0
ospf bfd enable
ospf bfd min-tx-interval 500 min-rx-interval 500 detect-multiplier 4

ospf 1
bfd all-interfaces enable
area 0.0.0.0
network 3.3.3.0 0.0.0.255
network 1.1.1.0 0.0.0.255

return
```

- DeviceB configuration file

```

sysname DeviceB

router id 2.2.2.2

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 2.2.2.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 3.3.3.2 255.255.255.0
ospf bfd enable
ospf bfd min-tx-interval 500 min-rx-interval 500 detect-multiplier 4

interface GigabitEthernet3/0/0
undo shutdown
```

```
ip address 172.16.1.1 255.255.255.0
#
ospf 1
bfd all-interfaces enable
area 0.0.0
network 3.3.3.0 0.0.0.255
network 2.2.2.0 0.0.0.255
network 172.16.1.0 0.0.0.255
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 2.2.2.1 255.255.255.0
ospf bfd enable
#
ospf 1
bfd all-interfaces enable
area 0.0.0
network 1.1.1.0 0.0.0.255
network 2.2.2.0 0.0.0.255
#
return
```

## Example for Configuring OSPF-BGP Synchronization

This section describes how to configure OSPF-BGP synchronization to minimize the impact of router restart on BGP services on the network.

### Networking Requirements

On the network shown in [Figure 1-136](#), all routers run BGP. An EBGP connection is set up between DeviceD and DeviceE. IBGP connections are set up between devices in AS 10, and OSPF is used as an IGP protocol.

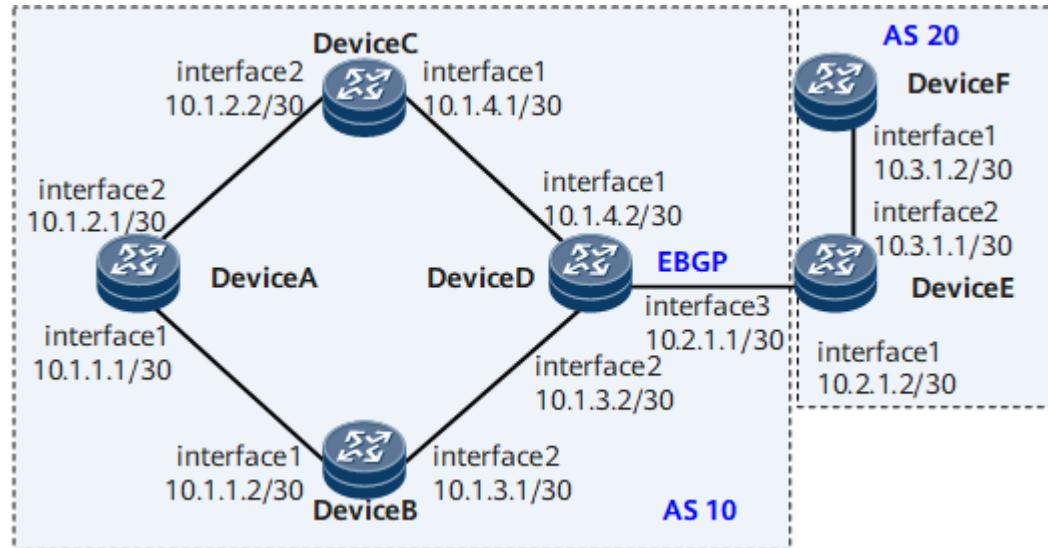
OSPF-BGP synchronization is required on DeviceB so that the restart of DeviceB does not interrupt the traffic from DeviceA to AS 20.

**Figure 1-136** Networking for configuring OSPF-BGP synchronization



#### NOTE

Interfaces 1 through 3 in this example represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable OSPF on DeviceA, DeviceB, DeviceC, and DeviceD (except 10.2.1.1/30) and specify the same area for all OSPF interfaces.
2. Set up IBGP connections between DeviceA, DeviceB, DeviceC, and DeviceD (except 10.2.1.1/30).
3. Configure the OSPF cost on DeviceC.
4. Set up EBGP connections between DeviceD and DeviceE.
5. Configure BGP to import direct routes and OSPF processes on DeviceD.
6. Configure BGP on DeviceE.

## Data Preparation

To complete the configuration, you need the following data:

- Data of DeviceA, including the router ID (1.1.1.1), number of the AS to which DeviceA belongs (10), OSPF process ID (1), network segment addresses of area 0 (10.1.1.0/30 and 10.1.2.0/30), and loopback0 IP address (1.1.1.1/32)
- Data of DeviceB, including the router ID (2.2.2.2), number of the AS to which DeviceB belongs (10), OSPF process ID (1), network segment addresses of area 0 (10.1.1.0/30 and 10.1.3.0/30), and loopback0 IP address (2.2.2.2/32)

- Data of DeviceC, including the router ID (3.3.3.3), number of the AS to which DeviceC belongs (10), OSPF process ID (1), network segment addresses of area 0 (10.1.2.0/30 and 10.1.4.0/30), and loopback0 IP address (3.3.3.3/32)
- Data of DeviceD, including the router ID (4.4.4.4), number of the AS to which DeviceD belongs (10), OSPF process ID (1), network segment addresses of area 0 (10.1.3.0/30 and 10.1.4.0/30), and loopback0 IP address (4.4.4.4/32)
- Data of DeviceE, including the router ID (5.5.5.5), number of the AS to which DeviceE belongs (20), and loopback0 IP address (5.5.5.5/32)

## Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic OSPF functions. For details, see [Example for Configuring Basic OSPF Functions](#).
- Step 3** Set up IBGP connections.

# Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] interface loopback 0
[~DeviceA-LoopBack0] ip address 1.1.1.1 32
[*DeviceA-LoopBack0] quit
[*DeviceA] bgp 10
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 2.2.2.2 as-number 10
[*DeviceA-bgp] peer 2.2.2.2 connect-interface LoopBack 0
[*DeviceA-bgp] peer 3.3.3.3 as-number 10
[*DeviceA-bgp] peer 3.3.3.3 connect-interface LoopBack 0
[*DeviceA-bgp] peer 4.4.4.4 as-number 10
[*DeviceA-bgp] peer 4.4.4.4 connect-interface LoopBack 0
[*DeviceA-bgp] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] interface loopback 0
[~DeviceB-LoopBack0] ip address 2.2.2.2 32
[*DeviceB-LoopBack0] quit
[*DeviceB] bgp 10
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 1.1.1.1 as-number 10
[*DeviceB-bgp] peer 1.1.1.1 connect-interface LoopBack 0
[*DeviceB-bgp] peer 3.3.3.3 as-number 10
[*DeviceB-bgp] peer 3.3.3.3 connect-interface LoopBack 0
[*DeviceB-bgp] peer 4.4.4.4 as-number 10
[*DeviceB-bgp] peer 4.4.4.4 connect-interface LoopBack 0
[*DeviceB-bgp] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
<DeviceC> system-view
[~DeviceC] interface loopback 0
[~DeviceC-LoopBack0] ip address 3.3.3.3 32
[*DeviceC-LoopBack0] quit
[*DeviceC] bgp 10
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 1.1.1.1 as-number 10
[*DeviceC-bgp] peer 1.1.1.1 connect-interface LoopBack 0
[*DeviceC-bgp] peer 2.2.2.2 as-number 10
```

```
[*DeviceC-bgp] peer 2.2.2.2 connect-interface LoopBack 0
[*DeviceC-bgp] peer 4.4.4.4 as-number 10
[*DeviceC-bgp] peer 4.4.4.4 connect-interface LoopBack 0
[*DeviceC-bgp] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
<DeviceD> system-view
[~DeviceD] interface loopback 0
[~DeviceD-LoopBack0] ip address 4.4.4.4 32
[*DeviceD-LoopBack0] quit
[*DeviceD] bgp 10
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 1.1.1.1 as-number 10
[*DeviceD-bgp] peer 1.1.1.1 connect-interface LoopBack 0
[*DeviceD-bgp] peer 2.2.2.2 as-number 10
[*DeviceD-bgp] peer 2.2.2.2 connect-interface LoopBack 0
[*DeviceD-bgp] peer 3.3.3.3 as-number 10
[*DeviceD-bgp] peer 3.3.3.3 connect-interface LoopBack 0
[*DeviceD-bgp] quit
[*DeviceD] commit
```

**Step 4** Set up EBGP connections.

# Configure DeviceD.

```
[~DeviceD] bgp 10
[*DeviceD-bgp] peer 10.2.1.2 as-number 20
[*DeviceD-bgp] import-route direct
[*DeviceD-bgp] import-route ospf 1
[*DeviceD-bgp] quit
[*DeviceD] commit
```

# Configure DeviceE.

```
[~DeviceE] bgp 20
[*DeviceE-bgp] peer 10.2.1.1 as-number 10
[*DeviceE-bgp] ipv4-family unicast
[*DeviceE-bgp-af-ipv4] network 10.3.1.0 30
[*DeviceE-bgp-af-ipv4] quit
[*DeviceE-bgp] commit
```

**Step 5** Configure the OSPF cost on DeviceC.

```
[~DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] ospf cost 2
[*DeviceC-GigabitEthernet1/0/0] quit
[~DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ospf cost 2
[*DeviceC-GigabitEthernet2/0/0] commit
[*DeviceC-GigabitEthernet2/0/0] quit
```



After the OSPF cost is set to 2 on DeviceC, DeviceA selects only DeviceB as the intermediate device to the network segment 10.2.1.0, and DeviceC becomes a backup of DeviceB.

# Display the routing table on DeviceA. As shown in the routing table, the route to the network segment 10.3.1.0 is learned through BGP, and the outbound interface is GigabitEthernet1/0/0.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 20 Routes : 20
Destination/Mask Proto Pre Cost Flags NextHop Interface
```

|                    |        |     |   |    |           |                             |
|--------------------|--------|-----|---|----|-----------|-----------------------------|
| 1.1.1.1/32         | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 1.1.1.255/32       | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 2.2.2.2/32         | OSPF   | 10  | 3 | D  | 10.1.1.2  | GigabitEthernet1/0/0        |
| 4.4.4.0/24         | BGP    | 255 | 0 | RD | 4.4.4.4   | GigabitEthernet1/0/0        |
| 4.4.4.4/32         | OSPF   | 10  | 3 | D  | 10.1.1.2  | GigabitEthernet1/0/0        |
| 5.5.5.0/24         | BGP    | 255 | 0 | RD | 10.2.1.2  | GigabitEthernet1/0/0        |
| 10.1.1.0/30        | Direct | 0   | 0 | D  | 10.1.1.1  | GigabitEthernet1/0/0        |
| 10.1.1.3/32        | Direct | 0   | 0 | D  | 10.1.1.1  | GigabitEthernet1/0/0        |
| 10.1.1.1/32        | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 10.1.1.255/32      | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 10.1.1.255/32      | Direct | 0   | 0 | D  | 10.1.1.2  | GigabitEthernet1/0/0        |
| 10.1.1.2/32        | Direct | 0   | 0 | D  | 10.1.1.2  | GigabitEthernet1/0/0        |
| 10.1.1.255/32      | Direct | 0   | 0 | D  | 10.1.1.2  | GigabitEthernet1/0/0        |
| 10.1.2.0/30        | Direct | 0   | 0 | D  | 10.1.2.1  | GigabitEthernet2/0/0        |
| 10.1.2.3/32        | Direct | 0   | 0 | D  | 10.1.2.1  | GigabitEthernet2/0/0        |
| 10.1.2.1/32        | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 10.1.2.255/32      | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 10.1.2.2/32        | Direct | 0   | 0 | D  | 10.1.2.2  | GigabitEthernet2/0/0        |
| 10.1.2.255/32      | Direct | 0   | 0 | D  | 10.1.2.2  | GigabitEthernet2/0/0        |
| 127.0.0.0/8        | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 127.0.0.1/32       | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| 127.0.0.255/32     | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |
| <b>10.3.1.0/30</b> | OSPF   | 10  | 2 | D  | 10.1.1.2  | <b>GigabitEthernet1/0/0</b> |
| 10.1.3.1/32        | BGP    | 255 | 0 | RD | 4.4.4.4   | GigabitEthernet1/0/0        |
| 10.1.4.0/30        | OSPF   | 10  | 3 | D  | 10.1.1.2  | GigabitEthernet1/0/0        |
|                    | OSPF   | 10  | 3 | D  | 10.1.2.2  | GigabitEthernet2/0/0        |
| 10.1.4.1/32        | BGP    | 255 | 0 | RD | 4.4.4.4   | GigabitEthernet1/0/0        |
| 10.2.1.0/30        | BGP    | 255 | 0 | RD | 4.4.4.4   | GigabitEthernet1/0/0        |
| 10.2.1.2/32        | BGP    | 255 | 0 | RD | 4.4.4.4   | GigabitEthernet1/0/0        |
| 10.3.1.0/30        | BGP    | 255 | 0 | RD | 4.4.4.4   | GigabitEthernet1/0/0        |
| 255.255.255.255/32 | Direct | 0   | 0 | D  | 127.0.0.1 | InLoopBack0                 |

# Display the routing table on DeviceB.

[~DeviceB] **display ip routing-table**

Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

| Routing Table: _public_ |        |             |      |       |           |                      |
|-------------------------|--------|-------------|------|-------|-----------|----------------------|
| Destinations : 19       |        | Routes : 19 |      |       |           |                      |
| Destination/Mask        | Proto  | Pre         | Cost | Flags | NextHop   | Interface            |
| 2.2.2.2/32              | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 2.2.2.255/32            | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 1.1.1.1/32              | OSPF   | 10          | 2    | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 4.4.4.0/24              | BGP    | 255         | 0    | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 4.4.4.4/32              | OSPF   | 10          | 2    | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 5.5.5.0/24              | BGP    | 255         | 0    | RD    | 10.2.1.2  | GigabitEthernet2/0/0 |
| 10.1.1.0/30             | Direct | 0           | 0    | D     | 10.1.1.2  | GigabitEthernet1/0/0 |
| 10.1.1.3/32             | Direct | 0           | 0    | D     | 10.1.1.2  | GigabitEthernet1/0/0 |
| 10.1.1.1/32             | Direct | 0           | 0    | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.1.255/32           | Direct | 0           | 0    | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.1.2/32             | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.1.255/32           | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.2.0/30             | OSPF   | 10          | 2    | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.3.0/30             | Direct | 0           | 0    | D     | 10.1.3.1  | GigabitEthernet2/0/0 |
| 10.1.3.3/32             | Direct | 0           | 0    | D     | 10.1.3.1  | GigabitEthernet2/0/0 |
| 10.1.3.1/32             | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.3.255/32           | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.3.2/32             | Direct | 0           | 0    | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.1.3.255/32           | Direct | 0           | 0    | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 127.0.0.0/8             | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32            | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.255/32          | Direct | 0           | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.4.0/30             | OSPF   | 10          | 2    | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.1.4.1/32             | BGP    | 255         | 0    | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.2.1.0/30             | BGP    | 255         | 0    | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.2.1.2/32             | BGP    | 255         | 0    | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |

|                    |        |     |   |             |                      |
|--------------------|--------|-----|---|-------------|----------------------|
| 10.3.1.0/30        | BGP    | 255 | 0 | RD 10.1.3.2 | GigabitEthernet2/0/0 |
| 255.255.255.255/32 | Direct | 0   | 0 | D 127.0.0.1 | InLoopBack0          |

As shown in the routing table, DeviceB learns the route to 10.3.1.0 through BGP, and the outbound interface is GE 2/0/0. The routes to 10.1.2.0 and 10.1.4.0 can be learned through OSPF. The costs of these routes are both 2.

**Step 6** Enable OSPF-BGP synchronization on DeviceB.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] stub-router on-startup
[*DeviceB-ospf-1] commit
```

**Step 7** Verify the configuration.

```
Restart router DeviceB.
```

NOTE

Exercise caution when running this command because it may lead to a temporary network crash. In addition, save the configuration file of the router before restarting it.

```
<DeviceB> reboot
System will reboot! Continue?[Y/N] y
```

```
Display the routing table on DeviceA. As shown in the routing table, BGP learns the route to 10.3.1.0, and the outbound interface is GigabitEthernet2/0/0.
```

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 20 Routes : 20
Destination/Mask Proto Pre Cost Flags NextHop Interface
1.1.1.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
2.2.2.2/32 OSPF 10 4 D 10.1.2.2 GigabitEthernet2/0/0
4.4.4.0/24 BGP 255 0 RD 4.4.4.4 GigabitEthernet2/0/0
4.4.4.4/32 OSPF 10 4 D 10.1.2.2 GigabitEthernet2/0/0
5.5.5.0/24 BGP 255 0 RD 10.2.1.2 GigabitEthernet2/0/0
10.1.1.0/30 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
10.1.1.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
10.1.1.2/32 Direct 0 0 D 10.1.1.2 GigabitEthernet1/0/0
10.1.2.0/30 Direct 0 0 D 10.1.2.1 GigabitEthernet2/0/0
10.1.2.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
10.1.2.2/32 Direct 0 0 D 10.1.2.2 GigabitEthernet2/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
10.1.3.0/30 OSPF 10 2 D 10.1.1.2 GigabitEthernet1/0/0
10.1.3.1/32 BGP 255 0 RD 4.4.4.4 GigabitEthernet2/0/0
10.1.4.0/30 OSPF 10 3 D 10.1.2.2 GigabitEthernet2/0/0
10.1.4.1/32 BGP 255 0 RD 4.4.4.4 GigabitEthernet2/0/0
10.2.1.0/30 BGP 255 0 RD 4.4.4.4 GigabitEthernet2/0/0
10.2.1.2/32 BGP 255 0 RD 4.4.4.4 GigabitEthernet2/0/0
10.3.1.0/30 BGP 255 0 RD 4.4.4.4 GigabitEthernet2/0/0
```

```
Display the routing table on DeviceB. As shown in the routing table, only OSPF routes exist in the routing table because IGP routes converge faster than BGP routes do. The costs of the OSPF routes are 65535.
```

```
[~DeviceB] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 15 Routes : 15
Destination/Mask Proto Pre Cost Flags NextHop Interface
```

|              |             |    |              |   |           |                      |
|--------------|-------------|----|--------------|---|-----------|----------------------|
| 1.1.1.1/32   | <b>OSPF</b> | 10 | <b>65536</b> | D | 10.1.1.1  | GigabitEthernet1/0/0 |
| 2.2.2.2/32   | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |
| 4.4.4.4/32   | <b>OSPF</b> | 10 | <b>65536</b> | D | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.1.1.0/30  | Direct      | 0  | 0            | D | 10.1.1.2  | GigabitEthernet1/0/0 |
| 10.1.1.1/32  | Direct      | 0  | 0            | D | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.1.2/32  | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |
| 10.1.2.0/30  | <b>OSPF</b> | 10 | <b>65536</b> | D | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.3.0/30  | Direct      | 0  | 0            | D | 10.1.3.1  | GigabitEthernet2/0/0 |
| 10.1.3.1/32  | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |
| 10.1.3.2/32  | Direct      | 0  | 0            | D | 10.1.3.2  | GigabitEthernet2/0/0 |
| 127.0.0.0/8  | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32 | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |
| 10.1.4.0/30  | <b>OSPF</b> | 10 | <b>65536</b> | D | 10.1.3.2  | GigabitEthernet2/0/0 |
| 127.0.0.0/8  | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32 | Direct      | 0  | 0            | D | 127.0.0.1 | InLoopBack0          |

# Display the routing table on DeviceB again.

[~DeviceB] **display ip routing-table**  
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

| Routing Table: _public_ |        |     |             |       |           |                      |
|-------------------------|--------|-----|-------------|-------|-----------|----------------------|
| Destinations : 19       |        |     | Routes : 19 |       |           |                      |
| Destination/Mask        | Proto  | Pre | Cost        | Flags | NextHop   | Interface            |
| 2.2.2.2/32              | Direct | 0   | 0           | D     | 127.0.0.1 | InLoopBack0          |
| 1.1.1.1/32              | OSPF   | 10  | 2           | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 4.4.4.0/24              | BGP    | 255 | 0           | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 4.4.4.4/32              | OSPF   | 10  | 2           | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 5.5.5.0/24              | BGP    | 255 | 0           | RD    | 10.2.1.2  | GigabitEthernet2/0/0 |
| 10.1.1.0/30             | Direct | 0   | 0           | D     | 10.1.1.2  | GigabitEthernet1/0/0 |
| 10.1.1.1/32             | Direct | 0   | 0           | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.1.2/32             | Direct | 0   | 0           | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.2.0/30             | OSPF   | 10  | 2           | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.3.0/30             | Direct | 0   | 0           | D     | 10.1.3.1  | GigabitEthernet2/0/0 |
| 10.1.3.1/32             | Direct | 0   | 0           | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.3.2/32             | Direct | 0   | 0           | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 127.0.0.0/8             | Direct | 0   | 0           | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32            | Direct | 0   | 0           | D     | 127.0.0.1 | InLoopBack0          |
| 10.1.4.0/30             | OSPF   | 10  | 2           | D     | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.1.4.1/32             | BGP    | 255 | 0           | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.2.1.0/30             | BGP    | 255 | 0           | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.2.1.2/32             | BGP    | 255 | 0           | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |
| 10.3.1.0/30             | BGP    | 255 | 0           | RD    | 10.1.3.2  | GigabitEthernet2/0/0 |

As shown in the routing table, BGP routes on DeviceB have converged, and the routing information is the same as that displayed before the restart.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

router id 1.1.1.1

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.252

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.252
#
```

```
interface LoopBack0
ip address 1.1.1.1 255.255.255.255
#
bgp 10
router-id 1.1.1.1
peer 2.2.2.2 as-number 10
peer 2.2.2.2 connect-interface LoopBack 0
peer 3.3.3.3 as-number 10
peer 3.3.3.3 connect-interface LoopBack 0
peer 4.4.4.4 as-number 10
peer 4.4.4.4 connect-interface LoopBack 0
#
ospf 1
area 0.0.0.0
network 1.1.1.1 0.0.0.0
network 10.1.1.0 0.0.0.3
network 10.1.2.0 0.0.0.3
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
router id 2.2.2.2
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.252
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.252
#
interface LoopBack0
ip address 2.2.2.2 255.255.255.255
#
bgp 10
router-id 2.2.2.2
peer 1.1.1.1 as-number 10
peer 1.1.1.1 connect-interface LoopBack 0
peer 3.3.3.3 as-number 10
peer 3.3.3.3 connect-interface LoopBack 0
peer 4.4.4.4 as-number 10
peer 4.4.4.4 connect-interface LoopBack 0
#
ospf 1
stub-router on-startup
area 0.0.0.0
network 10.1.1.0 0.0.0.3
network 10.1.3.0 0.0.0.3
network 2.2.2.2 0.0.0.0
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.1 255.255.255.252
ospf cost 2
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.252
ospf cost 2
```

```

interface LoopBack0
ip address 3.3.3.3 255.255.255.255

bgp 10
router-id 3.3.3.3
peer 1.1.1.1 as-number 10
peer 1.1.1.1 connect-interface LoopBack 0
peer 2.2.2.2 as-number 10
peer 2.2.2.2 connect-interface LoopBack 0
peer 4.4.4.4 as-number 10
peer 4.4.4.4 connect-interface LoopBack 0

ospf 1
area 0.0.0.0
network 10.1.2.0 0.0.0.3
network 10.1.4.0 0.0.0.3
network 3.3.3.3 0.0.0.0

return
```

- DeviceD configuration file

```

sysname DeviceD

router id 4.4.4.4

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.2 255.255.255.252

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.252

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.252

interface LoopBack0
ip address 4.4.4.4 255.255.255.255

bgp 10
router-id 4.4.4.4
peer 10.2.1.2 as-number 20
peer 1.1.1.1 as-number 10
peer 1.1.1.1 connect-interface LoopBack 0
peer 2.2.2.2 as-number 10
peer 2.2.2.2 connect-interface LoopBack 0
peer 3.3.3.3 as-number 10
peer 3.3.3.3 connect-interface LoopBack 0

ipv4-family unicast
undo synchronization
import-route direct
import-route ospf 1
peer 10.2.1.2 enable

ospf 1
area 0.0.0.0
network 4.4.4.4 0.0.0.0
network 10.1.3.0 0.0.0.3
network 10.1.4.0 0.0.0.3

return
```

- DeviceE configuration file

```

sysname DeviceE
#
```

```
router id 5.5.5.5
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.252
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.3.1.1 255.255.255.252
#
interface LoopBack0
ip address 5.5.5.5 255.255.255.255
#
bgp 20
router-id 5.5.5.5
peer 10.2.1.1 as-number 10
#
ipv4-family unicast
undo synchronization
network 10.3.1.0 255.255.255.252
peer 10.2.1.1 enable
#
return
```

## Example for Configuring Routing Loop Detection for Routes Imported from BGP to OSPF

This section describes how to configure routing loop detection for routes imported from BGP to OSPF.

### Networking Requirements

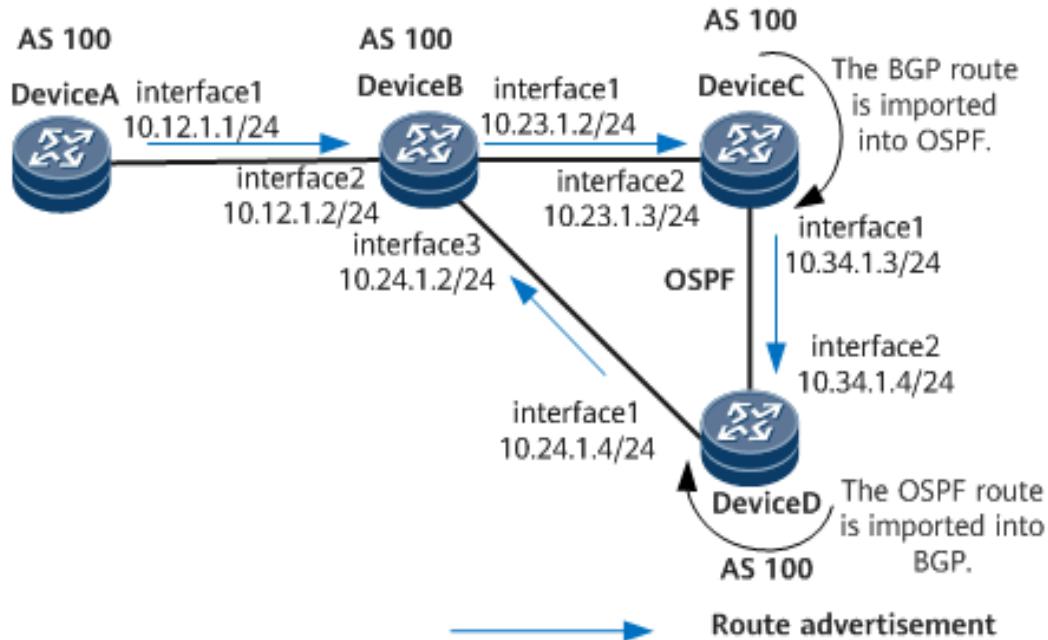
On the live network, OSPF routes can be imported to a BGP process for redistribution. In such a scenario, routing policies are usually configured on multiple devices to prevent routing loops. If routing policies are incorrectly configured on the devices that import routes, routing loops may occur. To prevent this problem, configure routing loop detection for the routes imported to OSPF.

On the network shown in [Figure 1-137](#), IBGP peer relationships are established between DeviceA and DeviceB, between DeviceB and DeviceC, between DeviceC and DeviceD, and between DeviceB and DeviceD; an OSPF process is configured on DeviceC and DeviceD. OSPF is configured to import BGP routes on DeviceC, and BGP is configured to import OSPF routes on DeviceD.

**Figure 1-137** Routing loop detection for routes imported from BGP to OSPF

#### NOTE

In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IP addresses for interfaces on each device.
2. Enable OSPF and BGP, and configure basic OSPF and BGP functions.
3. Configure route import to construct a routing loop.
4. Check whether a routing loop occurs.
5. Enable routing loop detection to check whether the routing loop is eliminated.

## Procedure

### Step 1 Assign an IP address to each interface.

DeviceA is used as an example.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ip address 10.12.1.1 24
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] commit
```

The configurations of other devices are similar to those of DeviceA. For configuration details, see [Configuration Files](#) in this section.

In addition, configure a static route on DeviceA to simulate a looped route.

```
[~DeviceA] ip route-static 10.0.0.0 255.255.255.255 NULL0
[*DeviceA] commit
```

**Step 2** Enable OSPF and BGP, and configure basic OSPF and BGP functions to implement intra-AS communication.

```
Enable BGP on DeviceA, and establish an IBGP peer relationship between
DeviceA and DeviceB.
```

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.11.1.1
[*DeviceA-bgp] peer 10.12.1.2 as-number 100
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 10.12.1.2 enable
[*DeviceA-bgp] quit
[*DeviceA] commit
```

```
Enable BGP on DeviceB and establish IBGP peer relationships between DeviceB
and DeviceA, between DeviceB and DeviceC, and between DeviceB and DeviceD.
```

```
[~DeviceB] bgp 100
[*DeviceB-bgp] router-id 2.22.2.2
[*DeviceB-bgp] peer 10.12.1.1 as-number 100
[*DeviceB-bgp] peer 10.23.1.3 as-number 100
[*DeviceB-bgp] peer 10.24.1.4 as-number 100
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] peer 10.12.1.1 enable
[*DeviceB-bgp-af-ipv4] peer 10.23.1.3 enable
[*DeviceB-bgp-af-ipv4] peer 10.24.1.4 enable
[*DeviceB-bgp-af-ipv4] peer 10.23.1.3 reflect-client
[*DeviceB-bgp] quit
[*DeviceB] commit
```

```
Enable BGP on DeviceC, and establish an IBGP peer relationship between
DeviceC and DeviceB.
```

```
[~DeviceC] bgp 100
[*DeviceC-bgp] router-id 3.33.3.3
[*DeviceC-bgp] peer 10.23.1.2 as-number 100
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] peer 10.23.1.2 enable
[*DeviceC-bgp] quit
[*DeviceC] commit
```

```
Enable BGP on DeviceD, and establish an IBGP peer relationship between
DeviceD and DeviceB.
```

```
[~DeviceD] bgp 100
[*DeviceD-bgp] router-id 4.44.4.4
[*DeviceD-bgp] peer 10.24.1.2 as-number 100
[*DeviceD-bgp] ipv4-family unicast
[*DeviceD-bgp-af-ipv4] peer 10.24.1.2 enable
[*DeviceD-bgp] quit
[*DeviceD] commit
```

```
Configure OSPF on DeviceC and DeviceD. DeviceC is used as an example.
```

```
[~DeviceC] ospf 1 router-id 3.33.3.3
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 10.34.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] quit
[*DeviceC-ospf-1] quit
[*DeviceC] commit
```

**Step 3** Configure route import.

```
Configure OSPF on DeviceC to import BGP routes.
```

```
[~DeviceC] ospf 1 router-id 3.33.3.3
[*DeviceC-ospf-1] import-route bgp permit-ibgp
[*DeviceC-ospf-1] quit
[*DeviceC] commit
```

# Configure BGP on DeviceD to import OSPF routes.

```
[~DeviceD] bgp 100
[*DeviceD-bgp] ipv4-family unicast
[*DeviceD-bgp-af-ipv4] import-route ospf 1
[*DeviceD-bgp] quit
[*DeviceD] commit
```

**Step 4** View the routing table on each device to check whether a routing loop occurs.

# Check BGP peer information on DeviceB.

```
[~DeviceB] display bgp peer
BGP local router ID : 2.22.2.2
Local AS number : 100
Total number of peers : 3 Peers in established state : 3

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
10.12.1.1 4 100 453 458 0 06:30:47 Established 1
10.23.1.3 4 100 452 458 0 06:30:46 Established 0
10.24.1.4 4 100 451 457 0 06:29:39 Established 3
```

# Check OSPF neighbor information on DeviceC.

```
[~DeviceC] display ospf peer
(M) Indicates MADJ neighbor

OSPF Process 1 with Router ID 3.33.3.3
Neighbors

Area 0.0.0 interface 10.34.1.3 (GigabitEthernet1/0/0)'s neighbors
Router ID: 4.44.4.4 Address: 10.34.1.4
State: Full Mode:Nbr is Master Priority: 1
DR: 10.34.1.4 BDR: 10.34.1.3 MTU: 0
Dead timer due in 31 sec
Retrans timer interval: 5
Neighbor is up for 06h28m21s
Neighbor Up Time : 2021-08-27 02:59:32
Authentication Sequence: [0]
```

# Check OSPF neighbor information on DeviceD.

```
[~DeviceD] display ospf peer
(M) Indicates MADJ neighbor

OSPF Process 1 with Router ID 4.44.4.4
Neighbors

Area 0.0.0 interface 10.34.1.4 (GigabitEthernet2/0/0)'s neighbors
Router ID: 3.33.3.3 Address: 10.34.1.3
State: Full Mode:Nbr is Slave Priority: 1
DR: 10.34.1.4 BDR: 10.34.1.3 MTU: 0
Dead timer due in 32 sec
Retrans timer interval: 5
Neighbor is up for 06h28m25s
Neighbor Up Time : 2021-08-27 02:59:32
Authentication Sequence: [0]
```

The preceding command outputs show that BGP peer relationships and OSPF neighbor relationships have been established between the devices.

# Check the BGP routing table on DeviceB.

```
[~DeviceB] display bgp routing-table 10.0.0.0
BGP local router ID : 2.22.2.2
Local AS number : 100
Paths: 2 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 10.0.0.0/32:
RR-client route.
From: 10.24.1.4 (4.44.4.4)
Route Duration: 0d00h00m52s
Relay IP Nexthop: 10.24.1.4
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.24.1.4
Qos information : 0x0
AS-path Nil, origin incomplete, MED 1, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Advertised to such 3 peers:
 10.23.1.3
 10.24.1.4
 10.12.1.1

BGP routing table entry information of 10.0.0.0/32:
From: 10.12.1.1 (1.11.1.1)
Route Duration: 0d22h53m22s
Relay IP Nexthop: 10.12.1.1
Relay IP Out-Interface: GigabitEthernet2/0/0
Original nexthop: 10.12.1.1
Qos information : 0x0
AS-path 10, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for
AS-Path
Not advertised to any peer yet
```

The preceding command output shows that DeviceB has learned the BGP route distributed by DeviceD.

# Check the BGP routing table of DeviceC.

```
[~DeviceC] display bgp routing-table 10.0.0.0
BGP local router ID : 3.33.3.3
Local AS number : 100
Paths: 1 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 10.0.0.0/32:
From: 10.23.1.2 (2.22.2.2)
Route Duration: 0d07h12m30s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface: NULL0
Original nexthop: 10.12.1.1
Qos information : 0x0
AS-path 10, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Originator: 1.11.1.1
Cluster list: 2.22.2.2
Not advertised to any peer yet
```

The preceding command output shows that DeviceC has learned the BGP route distributed by DeviceB.

# Check the routing table of DeviceD.

```
[~DeviceD] display ospf routing 10.0.0.0
OSPF Process 1 with Router ID 4.44.4.4

Destination : 10.0.0.0/32
AdverRouter : 3.33.3.3 Tag : 1
Cost : 1 Type : Type2
NextHop : 10.34.1.3 Interface : GigabitEthernet2/0/0
Priority : Medium Age : 01h31m18s
```

The preceding command output shows that DeviceD has learned the OSPF route distributed by DeviceC.

This means that a routing loop occurs among DeviceB, DeviceC, and DeviceD.

**Step 5** Enable routing loop detection on each device.

# Enable routing loop detection for routes imported to OSPF and BGP. DeviceA is used as an example.

```
[~DeviceA] route loop-detect ospf enable
[*DeviceA] route loop-detect bgp enable
[*DeviceA] commit
```

 **NOTE**

In the case of inter-protocol route import, if a routing protocol with a higher preference detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower preference is used for such processing, this route competes with the originally imported route during route selection. In this case, the routing loop can be eliminated. OSPF has a higher preference than BGP. Therefore, to eliminate the routing loop, BGP needs to reduce the preference of the corresponding route.

**Step 6** Check whether the routing loop is eliminated.

# Check the BGP routing table on DeviceB.

```
[~DeviceB] display bgp routing-table 10.0.0.0
BGP local router ID : 2.22.2.2
Local AS number : 100
Paths: 1 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 10.0.0.0/32:
From: 10.12.1.1 (1.1.1.1)
Route Duration: 1d00h10m02s
Relay IP Nexthop: 10.12.1.1
Relay IP Out-Interface: GigabitEthernet2/0/0
Original nexthop: 10.12.1.1
Qos information : 0x0
AS-path 10, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Advertised to such 2 peers:
 10.23.1.3
 10.24.1.4
```

The preceding command output shows that DeviceB has learned the route distributed by DeviceA and that DeviceB no longer prefers the route distributed by DeviceD. This means that the routing loop among DeviceB, DeviceC, and DeviceD is eliminated.

**----End**

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.12.1.1 255.255.255.0

bgp 100
router-id 1.1.1.1
private-4-byte-as enable
peer 10.12.1.2 as-number 100

ipv4-family unicast
undo synchronization
import-route static
peer 10.12.1.2 enable
#
```

```
ip route-static 10.0.0.0 255.255.255.255 NULL0
#
route loop-detect ospf enable
#
route loop-detect bgp enable
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.12.1.2 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.24.1.2 255.255.255.0
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.23.1.2 255.255.255.0
#
bgp 100
router-id 2.22.2.2
private-4-byte-as enable
peer 10.12.1.1 as-number 100
peer 10.23.1.3 as-number 100
peer 10.24.1.4 as-number 100
#
ipv4-family unicast
undo synchronization
peer 10.12.1.1 enable
peer 10.23.1.3 enable
peer 10.23.1.3 reflect-client
peer 10.24.1.4 enable
peer 10.24.1.4 reflect-client
#
route loop-detect ospf enable
#
route loop-detect bgp enable
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.34.1.3 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.23.1.3 255.255.255.0
#
bgp 100
router-id 3.33.3.3
private-4-byte-as enable
peer 10.23.1.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 10.23.1.2 enable
#
ospf 1 router-id 3.33.3.3
import-route bgp permit-ibgp
opaque-capability enable
area 0.0.0.0
```

```
network 10.34.1.0 0.0.0.255
#
route loop-detect ospf enable
#
route loop-detect bgp enable
#
return

● DeviceD configuration file

#
sysname DeviceD
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.34.1.4 255.255.255.0
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.24.1.4 255.255.255.0
#
bgp 100
router-id 4.44.4.4
private-4-byte-as enable
peer 10.24.1.2 as-number 100
#
ipv4-family unicast
undo synchronization
import-route ospf 1
peer 10.24.1.2 enable
#
ospf 1 router-id 4.44.4.4
opaque-capability enable
area 0.0.0.0
network 10.34.1.0 0.0.0.255
#
route loop-detect ospf enable
#
route loop-detect bgp enable
#
return
```

## Example for Configuring OSPF Delay Reporting to BGP-LS

Border Gateway Protocol-Link State (BGP-LS) can be used to summarize topology information collected by IGPs and send the information to the upper-layer controller.

## Networking Requirements

BGP-LS is a new method of collecting network topology information. The topology information discovered by IGPs is summarized and reported to an upper-layer controller through BGP. With powerful routing capabilities of BGP, BGP-LS has the following advantages:

- Lowers the requirements on the controller's computing and IGP capabilities.
- Facilitates route selection and computation on the controller by using BGP to summarize process or AS topology information and report the complete information to the controller.
- Requires only one routing protocol (BGP) to report topology information to the controller.

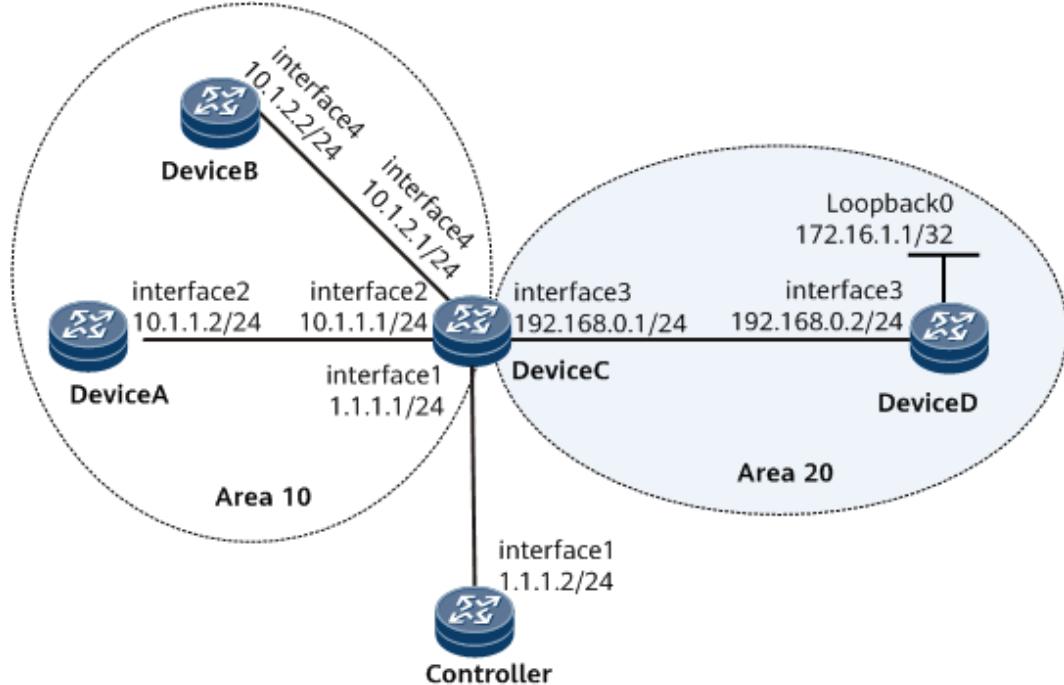
In [Figure 1-138](#), DeviceC is connected to the controller and reports topology information to the controller. DeviceA, DeviceB, DeviceC, and DeviceD use OSPF to

implement IP network interworking. The two interfaces connecting DeviceC and DeviceD belong to area 20, and the interfaces connecting DeviceA, DeviceB, and DeviceC belong to area 10.

**Figure 1-138** Configuring BGP-LS

 **NOTE**

In this example, interface1, interface2, interface3, and interface4 represent GigabitEthernet1/0/1, GigabitEthernet1/0/2, GigabitEthernet1/0/3, and GigabitEthernet1/0/4, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an IP address for each interface on each router.
2. Configure basic OSPF functions.
3. Deploy BGP-LS on DeviceC and the controller.
4. Configure DeviceA and DeviceC as the TWAMP sender and receiver, respectively.
5. Configure delay advertisement on DeviceA.

## Data Preparation

To complete the configuration, you need the following data:

- IP addresses and areas of interfaces on DeviceA, DeviceB, DeviceC, and DeviceD
- BGP-LS ID in OSPF on DeviceC
- BGP AS numbers, BGP-LS domain AS numbers, and BGP-LS domain IDs of DeviceC and the controller

## Procedure

**Step 1** Assign an IP address to each interface on each router. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic OSPF functions.

# Configure DeviceA.

```
[~DeviceA] ospf 1 router-id 1.1.1.1
[*DeviceA-ospf-1] area 0.0.0.10
[*DeviceA-ospf-1-area-0.0.0.10] quit
[*DeviceA-ospf-1] quit
[*DeviceA] interface gigabitethernet 1/0/2
[*DeviceA-GigabitEthernet1/0/2] ospf enable 1 area 0.0.0.10
[*DeviceA-GigabitEthernet1/0/2] ospf network-type p2p
[*DeviceA-GigabitEthernet1/0/2] commit
[~DeviceA-GigabitEthernet1/0/2] quit
```

# Configure DeviceB.

```
[~DeviceB] ospf 1 router-id 2.2.2.2
[*DeviceB-ospf-1] area 0.0.0.10
[*DeviceB-ospf-1-area-0.0.0.10] quit
[*DeviceB-ospf-1] quit
[*DeviceB] interface gigabitethernet 1/0/4
[*DeviceB-GigabitEthernet1/0/4] ospf enable 1 area 0.0.0.10
[*DeviceB-GigabitEthernet1/0/4] commit
[~DeviceB-GigabitEthernet1/0/4] quit
```

# Configure DeviceC.

```
[~DeviceC] ospf 1 router-id 3.3.3.3
[*DeviceC-ospf-1] area 0.0.0.10
[*DeviceC-ospf-1-area-0.0.0.10] quit
[*DeviceC-ospf-1] area 0.0.0.20
[*DeviceC-ospf-1-area-0.0.0.20] quit
[*DeviceC-ospf-1] quit
[*DeviceC] interface gigabitethernet 1/0/2
[*DeviceC-GigabitEthernet1/0/2] ospf enable 1 area 0.0.0.10
[*DeviceC-GigabitEthernet1/0/2] ospf network-type p2p
[*DeviceC-GigabitEthernet1/0/2] quit
[*DeviceC] interface gigabitethernet 1/0/3
[*DeviceC-GigabitEthernet1/0/3] ospf enable 1 area 0.0.0.20
[*DeviceC-GigabitEthernet1/0/3] quit
[*DeviceC] interface gigabitethernet 1/0/4
[*DeviceC-GigabitEthernet1/0/4] ospf enable 1 area 0.0.0.10
[*DeviceC-GigabitEthernet1/0/4] commit
[~DeviceC-GigabitEthernet1/0/4] quit
```

# Configure DeviceD.

```
[~DeviceD] ospf 1 router-id 4.4.4.4
[*DeviceD-ospf-1] area 0.0.0.20
[*DeviceD-ospf-1-area-0.0.0.20] quit
[*DeviceD-ospf-1] quit
[*DeviceD] interface gigabitethernet 1/0/3
[*DeviceD-GigabitEthernet1/0/3] ospf enable 1 area 20
[*DeviceD-GigabitEthernet1/0/3] quit
[*DeviceD] interface LoopBack0
```

```
[*DeviceD-LoopBack0] ospf enable 1 area 20
[*DeviceD-LoopBack0] commit
[~DeviceD-LoopBack0] quit
```

# Check OSPF routing information on each router. The following example uses the command output on DeviceC.

```
[~DeviceC] display ospf routing
```

```
OSPF Process 1 with Router ID 3.3.3.3
Routing Tables
```

Routing for Network

| Destination    | Cost | Type   | NextHop     | AdvRouter | Area     |
|----------------|------|--------|-------------|-----------|----------|
| 10.1.1.0/24    | 1    | Direct | 10.1.1.1    | 3.3.3.3   | 0.0.0.10 |
| 10.1.2.0/24    | 1    | Direct | 10.1.2.1    | 3.3.3.3   | 0.0.0.10 |
| 172.16.1.1/32  | 1    | Stub   | 192.168.0.2 | 4.4.4.4   | 0.0.0.20 |
| 192.168.0.0/24 | 1    | Direct | 192.168.0.1 | 3.3.3.3   | 0.0.0.20 |

Total Nets: 4

Intra Area: 4 Inter Area: 0 ASE: 0 NSSA: 0

**Step 3** Deploy BGP-LS on DeviceC and the controller.

# Enable OSPF topology advertisement on DeviceC.

```
[~DeviceC] ospf 1
[*DeviceC-ospf-1] bgp-ls enable
[*DeviceC-ospf-1] bgp-ls identifier 20
[*DeviceC-ospf-1] commit
[~DeviceC-ospf-1] quit
```

# Enable BGP-LS on DeviceC and establish a BGP-LS peer relationship with the controller.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] peer 1.1.1.2 as-number 100
[*DeviceC-bgp] link-state-family unicast
[*DeviceC-bgp-af-ls] peer 1.1.1.2 enable
[*DeviceC-bgp-af-ls] commit
[~DeviceC-bgp-af-ls] quit
[~DeviceC-bgp] quit
```

# Enable BGP-LS on the controller and establish a BGP-LS peer relationship with DeviceC.

```
[~Controller] bgp 100
[*Controller-bgp] peer 1.1.1.1 as-number 100
[*Controller-bgp] link-state-family unicast
[*Controller-bgp-af-ls] peer 1.1.1.1 enable
[*Controller-bgp-af-ls] commit
[~Controller-bgp-af-ls] quit
[~Controller-bgp] quit
```

**Step 4** Configure DeviceA and DeviceC as the TWAMP sender and receiver, respectively.

# Configure DeviceA as the TWAMP client and sender.

```
[~DeviceA] nqa twamp-light
[*DeviceA-twamp-light] client
[*DeviceA-twamp-light-client] test-session 1 sender-ip 10.1.1.2 reflector-ip 10.1.1.1 sender-port 862
reflector-port 862
[*DeviceA-twamp-light-client] test-session 1 bind interface GigabitEthernet 1/0/2
[*DeviceA-twamp-light-client] quit
[*DeviceA-twamp-light] sender
[*DeviceA-twamp-light-sender] test start-continual test-session 1 period 1000
[*DeviceA-twamp-light-sender] commit
[~DeviceA-twamp-light-sender] quit
[~DeviceA-twamp-light] quit
```

# Configure DeviceC as the TWAMP receiver.

```
[~DeviceC] nqa twamp-light
[*DeviceC-twamp-light] responder
[*DeviceC-twamp-light-responder] test-session 1 local-ip 10.1.1.1 remote-ip 10.1.1.2 local-port 862
remote-port 862
[*DeviceC-twamp-light-responder] commit
[~DeviceC-twamp-light-responder] quit
[~DeviceC-twamp-light] quit
```

**Step 5** Configure delay advertisement on DeviceA.

```
[~DeviceA] ospf 1 router-id 1.1.1.1
[*DeviceA-ospf-1] metric-delay advertisement enable
[*DeviceA-ospf-1] commit
```

**Step 6** Verify the configuration.

# Display information about BGP-LS peers and their status on DeviceC.

```
[~DeviceC] display bgp link-state unicast peer
BGP local router ID : 1.1.1.1
Local AS number : 100
Total number of peers : 1 Peers in established state : 1

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
1.1.1.2 4 100 5 9 0 00:00:55 Established 0
```

# Check the delay sub-TLVs carried in OSPF Opaque LSAs on DeviceC.

```
[~DeviceC] display ospf lsdb opaque-area
OSPF Process 1 with Router ID 3.3.3.3
Area: 0.0.0.10
Link State Database

Type : Opq-Area
Ls id : 8.0.0.0
Adv rtr : 1.1.1.1
Ls age : 55
Len : 48
Options : E
seq# : 80000002
chksum : 0x365e
Opaque Type: 8
Opaque Id: 0
OSPFv2 Extended Link Opaque LSA TLV information:
OSPFv2 Extended Link TLV:
Link Type: P-2-P
Link ID: 3.3.3.3
Link Data: 10.1.1.2
Min/Max Unidirectional Link Delay Sub-TLV:
Anomalous (A) Bit: 0
Min Delay: 786 (us)
```

# Check BGP-LS routes on DeviceC.

```
[~DeviceC] display bgp link-state unicast routing-table [LINK][OSPFv2][IDENTIFIER20][LOCAL[as100]
[bgp-ls-identifier1.1.1.1][ospf-area-id0.0.0.10][igp-router-id1.1.1.1]][REMOTE[as100][bgp-ls-
identifier1.1.1.1][ospf-area-id0.0.0.10][igp-router-id3.3.3.3]][LINK[if-address10.1.1.2][peer-
address10.1.1.1][if-address::][peer-address::]]
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete

BGP local router ID : 1.1.1.1
Local AS number : 100
Paths: 1 available, 1 best, 1 select
BGP routing table entry information of [LINK][OSPFv2][IDENTIFIER20][LOCAL[as100][bgp-ls-
```

```
identifier1.1.1.1][ospf-area-id0.0.0.10][igp-router-id1.1.1.1]][REMOTE[as100][bgp-ls-identifier1.1.1.1][ospf-area-id0.0.0.10][igp-router-id3.3.3.3]][LINK[if-address10.1.1.2][peer-address10.1.1.1][if-address::][peer-address::]]:
Imported route.
From: 0.0.0.0 (0.0.0.0)
Route Duration: 0d00h09m59s
Direct Out-interface:
Original nexthop: 0.0.0.0
AS-path Nil, origin incomplete, MED 0, pref-val 0, valid, local, best, select, pre 255
Link-state Route Type: LINK
Protocol: OSPFv2
Identifier: 20
Local Node Descriptor:
AS Number: 100
BGP Identifier: 1.1.1.1
OSPF Area ID: 0.0.0.10
IGP Router ID: 1.1.1.1
Remote Node Descriptor:
AS Number: 100
BGP Identifier: 1.1.1.1
OSPF Area ID: 0.0.0.10
IGP Router ID: 3.3.3.3
Link Descriptor:
IPv4 interface address: 10.1.1.2
IPv4 neighbor address: 10.1.1.1
IPv6 interface address: ::
IPv6 neighbor address: ::
Link-state attribute:
IPv4 Router ID of Local Node:
IPv6 Router ID of Local Node:
IPv4 Router ID of Remote Node:
IPv6 Router ID of Remote Node:
Administrative group: 0x0
Maximum link bandwidth(kbits/sec): 0
Maximum reservable link bandwidth(kbits/sec): 0
Maximum Unreserved bandwidth(kbits/sec): 0 0 0 0 0 0 0
TE Default Metric: 0
Link Protection Type: Null
MPLS Protocol Mask: 0x0
IGP Metric: 1
Shared Risk Link Group:
Min/Max Unidirectional Link Delay(microseconds): 912/5503
Opaque link Properties:
Link Name:
Adjacency Segment Identifier(Flags/Weight/SID):
LAN Adjacency Segment Identifier(Flags/Weight/System-ID/SID):
SRv6 End.X SID(EndPoint Behavior/Flags/Algorithm/Weight/SID)(SID Structure):
SRv6 LAN End.X SID(EndPoint Behavior/Flags/Algorithm/Weight/Neighbor ID/SID)(SID Structure):
MSD([MSD Type, MSD Value]):
Advertised to such 1 peers:
1.1.1.2
```

The preceding command output shows that DeviceC obtains the topology information on the whole OSPF network. DeviceC can use BGP-LS routes to report the topology information to its BGP-LS peer (the controller).

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

ospf 1 router-id 1.1.1.1
metric-delay advertisement enable
area 0.0.0.10
#
```

```
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.1.2 255.255.255.0
ospf network-type p2p
ospf enable 1 area 0.0.0.10
#
nqa twamp-light
client
 test-session 1 sender-ip 10.1.1.2 reflector-ip 10.1.1.1 sender-port 862 reflector-port 862
 test-session 1 bind interface GigabitEthernet 1/0/2
sender
 test start-continual test-session 1 period 1000
#
return
```

- DeviceB configuration file

```

sysname DeviceB
#
ospf 1 router-id 2.2.2.2
area 0.0.0.10
#
interface GigabitEthernet1/0/4
undo shutdown
ip address 10.1.2.2 255.255.255.0
ospf enable 1 area 0.0.0.10
#
return
```

- DeviceC configuration file

```

sysname DeviceC
#
ospf 1 router-id 3.3.3.3
bgp-ls enable
bgp-ls identifier 20
area 0.0.0.10
area 0.0.0.20
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 1.1.1.1 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.1.1 255.255.255.0
ospf network-type p2p
ospf enable 1 area 0.0.0.10
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 192.168.0.1 255.255.255.0
ospf enable 1 area 0.0.0.20
#
interface GigabitEthernet1/0/4
undo shutdown
ip address 10.1.2.1 255.255.255.0
ospf enable 1 area 0.0.0.10
#
bgp 100
peer 1.1.1.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.1.2 enable
#
link-state-family unicast
peer 1.1.1.2 enable
#
nqa twamp-light
```

```
responder
test-session 1 local-ip 10.1.1.1 remote-ip 10.1.1.2 local-port 862 remote-port 862
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
ospf 1 router-id 4.4.4.4
area 0.0.0.20
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 192.168.0.2 255.255.255.0
ospf enable 1 area 0.0.0.20
#
interface LoopBack0
ip address 172.16.1.1 255.255.255.255
ospf enable 1 area 0.0.0.20
#
return
```

- Controller configuration file

```
#
sysname Controller
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 1.1.1.2 255.255.255.0
#
bgp 100
peer 1.1.1.1 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.1.1 enable
#
link-state-family unicast
peer 1.1.1.1 enable
#
return
```

## Example for Configuring OSPF Multi-Area Adjacency

This section provides an example for configuring OSPF multi-area adjacency, which includes configuration of enabling OSPF on each device, physical interface, and OSPF multi-area adjacency interface.

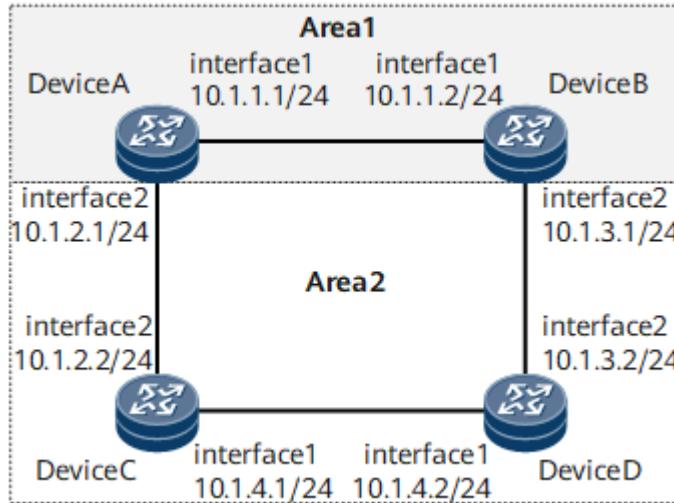
## Networking Requirements

In [Figure 1-139](#), all routers run OSPF, and the AS consists of two areas: area 1 and area 2. The link between DeviceA and DeviceB in area 1 is a high-speed link. Because intra-area links take precedence over inter-area links during OSPF route selection, traffic from DeviceA to DeviceB in area 2 is forwarded along the low-speed link of DeviceA->DeviceC->DeviceD->DeviceB, even though high-speed link between DeviceA and DeviceB exists. It is required that traffic from DeviceA to DeviceB in area 2 be forwarded along the high-speed link between DeviceA and DeviceB. In this case, configure OSPF multi-area adjacency on DeviceA and DeviceB and add their multi-area adjacency interfaces to area 2.

**Figure 1-139** Networking for configuring OSPF multi-area adjacency

 **NOTE**

Interfaces 1 and 2 in this example represent GE1/0/0 and GE2/0/0, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable OSPF on each router.
2. Enable OSPF on physical interfaces.
3. Enable OSPF on multi-area adjacency interfaces.

## Data Preparation

To complete the configuration, you need the following data:

- OSPF process ID: 1
- OSPF areas: area 1 and area 2

## Procedure

**Step 1** Configure an IP address for each interface. The configuration details are not provided here.

**Step 2** Configure basic OSPF functions.

```
Configure DeviceA.
```

```
[~DeviceA] ospf 1
[*DeviceA-ospf-1] area 1
[*DeviceA-ospf-1-area-0.0.0.1] quit
[*DeviceA-ospf-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ospf enable 1 area 1
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] ospf 1
[*DeviceA-ospf-1] area 2
[*DeviceA-ospf-1-area-0.0.0.2] quit
[*DeviceA-ospf-1] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] ospf enable 1 area 2
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 1
[*DeviceB-ospf-1-area-0.0.0.1] quit
[*DeviceB-ospf-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] ospf enable 1 area 1
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] ospf 1
[*DeviceB-ospf-1] area 2
[*DeviceB-ospf-1-area-0.0.0.2] quit
[*DeviceB-ospf-1] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] ospf enable 1 area 2
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
[~DeviceC] ospf 1
[*DeviceC-ospf-1] area 2
[*DeviceC-ospf-1-area-0.0.0.2] quit
[*DeviceC-ospf-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] ospf enable 1 area 2
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ospf enable 1 area 2
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] ospf 1
[*DeviceD-ospf-1] area 2
[*DeviceD-ospf-1-area-0.0.0.2] quit
[*DeviceD-ospf-1] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] ospf enable 1 area 2
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] ospf enable 1 area 2
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] commit
```

# Run the **display ospf peer brief** command to check brief information about OSPF neighbors. DeviceA is used as an example. The following command output shows that the OSPF neighbor relationships between DeviceA and DeviceB and between DeviceA and DeviceC are established.

```
[~DeviceA] display ospf peer brief
(M) Indicates MADJ neighbor
```

```
OSPF Process 1 with Router ID 1.1.1.1
Peer Statistic Information
Total number of peer(s): 2
Peer(s) in full state: 2

Area Id Interface Neighbor id State
0.0.0.1 GigabitEthernet1/0/0 10.1.1.2 Full
0.0.0.2 GigabitEthernet2/0/0 10.1.1.1 Full
```

# Run the **display ip routing-table** command to check information about the IP routing table. DeviceA is used as an example. The following command output shows that the outbound interface of the route destined for 1.1.1.1 is GigabitEthernet 2/0/0.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Destinations : 13 Routes : 13
Destination/Mask Proto Pre Cost Flags NextHop Interface
1.1.1.1/32 OSPF 10 3 D 10.1.2.2 GigabitEthernet2/0/0
10.1.1.0/24 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
10.1.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
10.1.2.0/24 Direct 0 0 D 10.1.2.1 GigabitEthernet2/0/0
10.1.2.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
10.1.2.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
10.1.3.0/24 OSPF 10 3 D 10.1.2.2 GigabitEthernet2/0/0
10.1.4.0/24 OSPF 10 2 D 10.1.2.2 GigabitEthernet2/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

The preceding command output shows that traffic from routerA to routerB in area 2 is forwarded along the low-speed link of routerA -> routerC -> routerD -> routerB.

### Step 3 Enable OSPF on multi-area adjacency interfaces.

# Configure DeviceA.

```
[~DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ospf enable multi-area 2
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[~DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] ospf enable multi-area 2
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] commit
```

### Step 4 Verify the configuration.

# Run the **display ospf peer brief** command to check brief information about OSPF neighbors. DeviceA is used as an example. The following command output shows that the OSPF neighbor relationships between DeviceA and DeviceB and between DeviceA and DeviceC are established and that an OSPF multi-area adjacency is established between DeviceA and DeviceB.

```
[~DeviceA] display ospf peer brief
```

(M) Indicates MADJ neighbor

| OSPF Process 1 with Router ID 1.1.1.1 |                      |             |       |  |
|---------------------------------------|----------------------|-------------|-------|--|
| Peer Statistic Information            |                      |             |       |  |
| Total number of peer(s): 3            |                      |             |       |  |
| Peer(s) in full state: 3              |                      |             |       |  |
| Area Id                               | Interface            | Neighbor id | State |  |
| 0.0.0.1                               | GigabitEthernet1/0/0 | 10.1.1.2    | Full  |  |
| 0.0.0.2                               | GigabitEthernet1/0/0 | 10.1.1.2(M) | Full  |  |
| 0.0.0.2                               | GigabitEthernet2/0/0 | 10.1.1.1    | Full  |  |

# Run the **display ip routing-table** command to check information about the IP routing table. DeviceA is used as an example. The following command output shows that the outbound interface of the route destined for 1.1.1.1 is GigabitEthernet 1/0/0.

| [~DeviceA] display ip routing-table                                                    |        |     |      |       |           |                      |
|----------------------------------------------------------------------------------------|--------|-----|------|-------|-----------|----------------------|
| Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route |        |     |      |       |           |                      |
| Routing Table : _public_                                                               |        |     |      |       |           |                      |
| Destinations : 13 Routes : 13                                                          |        |     |      |       |           |                      |
| Destination/Mask                                                                       | Proto  | Pre | Cost | Flags | NextHop   | Interface            |
| 1.1.1.1/32                                                                             | OSPF   | 10  | 1    | D     | 10.1.1.2  | GigabitEthernet1/0/0 |
| 10.1.1.0/24                                                                            | Direct | 0   | 0    | D     | 10.1.1.1  | GigabitEthernet1/0/0 |
| 10.1.1.1/32                                                                            | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 10.1.1.255/32                                                                          | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 10.1.2.0/24                                                                            | Direct | 0   | 0    | D     | 10.1.2.1  | GigabitEthernet2/0/0 |
| 10.1.2.1/32                                                                            | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 10.1.2.255/32                                                                          | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 10.1.3.0/24                                                                            | OSPF   | 10  | 2    | D     | 10.1.1.2  | GigabitEthernet1/0/0 |
| 10.1.4.0/24                                                                            | OSPF   | 10  | 2    | D     | 10.1.2.2  | GigabitEthernet2/0/0 |
| 127.0.0.0/8                                                                            | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32                                                                           | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.255.255.255/32                                                                     | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 255.255.255.255/32                                                                     | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |

The preceding command output shows that traffic from routerA to routerB in area 2 is forwarded along the high-speed link between routerA and routerB.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
ospf enable 1 area 0.0.0.1
ospf enable multi-area 0.0.0.2

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
ospf enable 1 area 0.0.0.2

ospf 1
area 0.0.0.1
area 0.0.0.2

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
ospf enable 1 area 0.0.0.1
ospf enable multi-area 0.0.0.2

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0
ospf enable 1 area 0.0.0.2

interface LoopBack0
ip address 1.1.1.1 255.255.255.255
ospf enable 1 area 0.0.0.2

ospf 1
area 0.0.0.1
area 0.0.0.2

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.1 255.255.255.0
ospf enable 1 area 0.0.0.2

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
ospf enable 1 area 0.0.0.2

ospf 1
area 0.0.0.2

return
```

- DeviceD configuration file

```

sysname DeviceD

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.2 255.255.255.0
ospf enable 1 area 0.0.0.2

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
ospf enable 1 area 0.0.0.2

ospf 1
area 0.0.0.2

return
```

## Example for Configuring an OSPF Sham Link

This section provides an example for configuring an OSPF sham link so that traffic between sites of the same VPN in the same OSPF area is forwarded through the OSPF intra-area route over the BGP/MPLS IP VPN backbone network.

## Networking Requirements

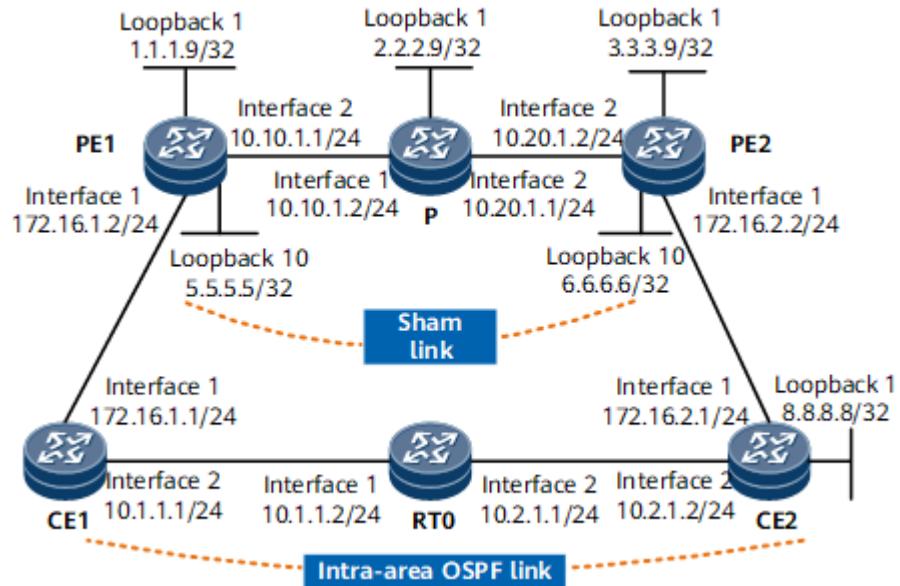
In [Figure 1-140](#), CE1 and CE2 reside in the same OSPF area, belong to VPN1, and are connected to PE1 and PE2, respectively. In this example, the cost of each link is 1.

It is required that OSPF run between each CE and its connected PE and the VPN traffic between CE1 and CE2 be forwarded over the MPLS backbone network.

**Figure 1-140** Networking for configuring an OSPF sham link

 NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



## Precautions

To improve security, OSPF area authentication or interface authentication is recommended. For details, see "Improving OSPF Network Security". OSPF area authentication is used as an example. For details, see Example for Configuring Basic OSPF Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an MP-IBGP peer relationship between PEs and configure OSPF between each CE and its connected PE.
2. Create a VPN instance on each PE and bind the VPN instance to the interface connected to the corresponding CE.
3. Create an OSPF sham link between PEs.
4. Adjust the cost of forwarding interfaces over the user network to ensure that the cost of the OSPF route used to forward the traffic over the user network is greater than that of the sham link.

## Data Preparation

To complete the configuration, you need the following data:

- MPLS LSR IDs of the PEs and P
- VPN instance name, RD, and VPN target of each PE
- Data used to configure OSPF (The OSPF process running on the backbone network, the OSPF process running on the user network, and the OSPF process connecting each PE to the corresponding CE are different.)
- Cost of the sham link and the cost of the OSPF route used to forward the traffic over the user network

## Procedure

### Step 1 Configure OSPF on the user network.

Configure OSPF on CE1, RT0, and CE2 and advertise network segments of their interfaces.

# Configure CE1.

```
<HUAWEI> system-view
[~HUAWEI] sysname CE1
[*HUAWEI] commit
[~CE1] interface GigabitEthernet2/0/0
[~CE1-GigabitEthernet2/0/0] ip address 10.1.1.1 24
[*CE1-GigabitEthernet2/0/0] quit
[~CE1] interface GigabitEthernet1/0/0
[~CE1-GigabitEthernet1/0/0] ip address 172.16.1.1 24
[*CE1-GigabitEthernet1/0/0] quit
[*CE1] ospf
[*CE1-ospf-1] area 0
[*CE1-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
[*CE1-ospf-1-area-0.0.0.0] network 172.16.1.0 0.0.0.255
[*CE1-ospf-1-area-0.0.0.0] quit
[*CE1-ospf-1] quit
[*CE1] commit
```

# Configure RT0.

```
<HUAWEI> system-view
[~HUAWEI] sysname RT0
[*HUAWEI] commit
[~RT0] interface GigabitEthernet1/0/0
[~RT0-GigabitEthernet1/0/0] ip address 10.1.1.2 24
[*RT0-GigabitEthernet1/0/0] quit
[~RT0] interface GigabitEthernet2/0/0
[~RT0-GigabitEthernet2/0/0] ip address 10.2.1.1 24
[*RT0-GigabitEthernet2/0/0] quit
[*RT0] ospf
[*RT0-ospf-1] area 0
[*RT0-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
[*RT0-ospf-1-area-0.0.0.0] network 10.2.1.0 0.0.0.255
[*RT0-ospf-1-area-0.0.0.0] quit
[*RT0-ospf-1] quit
[*RT0] commit
```

# Configure CE2.

```
<HUAWEI> system-view
[~HUAWEI] sysname CE2
[*HUAWEI] commit
[~CE2] interface GigabitEthernet2/0/0
[~CE2-GigabitEthernet2/0/0] ip address 10.2.1.2 24
```

```
[*CE2-GigabitEthernet2/0/0] quit
[*CE2] interface GigabitEthernet1/0/0
[*CE2-GigabitEthernet1/0/0] ip address 172.16.2.1 24
[*CE2-GigabitEthernet1/0/0] quit
[*CE2] ospf
[*CE2-ospf-1] area 0
[*CE2-ospf-1-area-0.0.0.0] network 10.2.1.0 0.0.0.255
[*CE2-ospf-1-area-0.0.0.0] network 172.16.2.0 0.0.0.255
[*CE2-ospf-1-area-0.0.0.0] quit
[*CE2-ospf-1] quit
[*CE2] commit
```

**Step 2** Configure basic BGP/MPLS IP VPN functions on the backbone network, including an IGP (OSPF) on the backbone network, MPLS and LDP on the backbone network, and an MP-IBGP peer relationship between PEs.

# Configure PE1.

```
<HUAWEI> system-view
[~HUAWEI] sysname PE1
[~PE1] interface loopback 1
[~PE1-LoopBack1] ip address 1.1.1.9 32
[*PE1-LoopBack1] quit
[*PE1] mpls lsr-id 1.1.1.9
[*PE1] mpls
[*PE1-mpls] quit
[*PE1] mpls ldp
[*PE1-mpls-ldp] quit
[*PE1] interface GigabitEthernet2/0/0
[*PE1-GigabitEthernet2/0/0] ip address 10.10.1.1 24
[*PE1-GigabitEthernet2/0/0] mpls
[*PE1-GigabitEthernet2/0/0] mpls ldp
[*PE1-GigabitEthernet2/0/0] quit
[*PE1] ospf
[*PE1-ospf-1] area 0
[*PE1-ospf-1-area-0.0.0.0] network 1.1.1.9 0.0.0.0
[*PE1-ospf-1-area-0.0.0.0] network 10.10.1.0 0.0.0.255
[*PE1-ospf-1-area-0.0.0.0] quit
[*PE1-ospf-1] quit
[*PE1] bgp 100
[*PE1-bgp] peer 3.3.3.9 as-number 100
[*PE1-bgp] peer 3.3.3.9 connect-interface loopback 1
[*PE1-bgp] ipv4-family vpngv4
[*PE1-bgp-af-vpngv4] peer 3.3.3.9 enable
[*PE1-bgp-af-vpngv4] quit
[*PE1-bgp] quit
[*PE1] commit
```

# Configure the P.

```
<HUAWEI> system-view
[~HUAWEI] sysname P
[~P] interface loopback 1
[~P-LoopBack1] ip address 2.2.2.9 32
[*P-LoopBack1] quit
[*P] mpls lsr-id 2.2.2.9
[*P] mpls
[*P-mpls] quit
[*P] mpls ldp
[*P-mpls-ldp] quit
[*P] interface GigabitEthernet1/0/0
[*P-GigabitEthernet1/0/0] ip address 10.10.1.2 24
[*P-GigabitEthernet1/0/0] mpls
[*P-GigabitEthernet1/0/0] mpls ldp
[*P-GigabitEthernet1/0/0] quit
[*P] interface GigabitEthernet2/0/0
[*P-GigabitEthernet2/0/0] ip address 10.20.1.1 24
[*P-GigabitEthernet2/0/0] mpls
[*P-GigabitEthernet2/0/0] mpls ldp
```

```
[*P-GigabitEthernet2/0/0] quit
[*P] ospf
[*P-ospf-1] area 0
[*P-ospf-1-area-0.0.0] network 2.2.2.9 0.0.0.0
[*P-ospf-1-area-0.0.0] network 10.10.1.0 0.0.0.255
[*P-ospf-1-area-0.0.0] network 10.20.1.0 0.0.0.255
[*P-ospf-1-area-0.0.0] quit
[*P-ospf-1] quit
[*P] commit
```

# Configure PE2.

```
<HUAWEI> system-view
[~HUAWEI] sysname PE2
[~PE2] interface loopback 1
[~PE2-LoopBack1] ip address 3.3.3.9 32
[*PE2-LoopBack1] quit
[*PE2] mpls lsr-id 3.3.3.9
[*PE2] mpls
[*PE2-mpls] quit
[*PE2] mpls ldp
[*PE2-mpls-ldp] quit
[*PE2] interface GigabitEthernet2/0/0
[*PE2-GigabitEthernet2/0/0] ip address 10.20.1.2 24
[*PE2-GigabitEthernet2/0/0] mpls
[*PE2-GigabitEthernet2/0/0] mpls ldp
[*PE2-GigabitEthernet2/0/0] quit
[*PE2] ospf
[*PE2-ospf-1] area 0
[*PE2-ospf-1-area-0.0.0] network 3.3.3.9 0.0.0.0
[*PE2-ospf-1-area-0.0.0] network 10.20.1.0 0.0.0.255
[*PE2-ospf-1-area-0.0.0] quit
[*PE2-ospf-1] quit
[*PE2] bgp 100
[*PE2-bgp] peer 1.1.1.9 as-number 100
[*PE2-bgp] peer 1.1.1.9 connect-interface loopback 1
[*PE2-bgp] ipv4-family vpnv4
[*PE2-bgp-af-vpnv4] peer 1.1.1.9 enable
[*PE2-bgp-af-vpnv4] quit
[*PE2-bgp] quit
[*PE2] commit
```

After completing the configurations, PE1 and PE2 learn the route to each other's loopback interface and establish an MP-IBGP peer relationship.

**Step 3** Configure the connection between each PE and the corresponding CE, with OSPF running between them.

# Configure PE1.

```
[~PE1] ip vpn-instance vpn1
[*PE1-vpn-instance-vpn1] ipv4-family
[*PE1-vpn-instance-vpn1-af-ipv4] route-distinguisher 100:1
[*PE1-vpn-instance-vpn1-af-ipv4] vpn-target 1:1
[*PE1-vpn-instance-vpn1-af-ipv4] quit
[*PE1-vpn-instance-vpn1] quit
[*PE1] interface gigabitethernet 1/0/0
[*PE1-GigabitEthernet1/0/0] ip binding vpn-instance vpn1
[*PE1-GigabitEthernet1/0/0] ip address 172.16.1.2 24
[*PE1-GigabitEthernet1/0/0] quit
[*PE1] ospf 100 vpn-instance vpn1
[*PE1-ospf-100] domain-id 10
[*PE1-ospf-100] import-route bgp
[*PE1-ospf-100] area 0
[*PE1-ospf-100-area-0.0.0] network 172.16.1.0 0.0.0.255
[*PE1-ospf-100-area-0.0.0] quit
[*PE1-ospf-100] quit
[*PE1] bgp 100
[*PE1-bgp] ipv4-family vpn-instance vpn1
```

```
[*PE1-bgp-vpn1] import-route direct
[*PE1-bgp-vpn1] import-route ospf 100
[*PE1-bgp-vpn1] quit
[*PE1-bgp] quit
[*PE1] commit
```

# Configure PE2.

```
[~PE2] ip vpn-instance vpn1
[*PE2-vpn-instance-vpn1] ipv4-family
[*PE2-vpn-instance-vpn1-af-ipv4] route-distinguisher 100:2
[*PE2-vpn-instance-vpn1-af-ipv4] vpn-target 1:1
[*PE2-vpn-instance-vpn1-af-ipv4] quit
[*PE2-vpn-instance-vpn1] quit
[*PE2] interface GigabitEthernet1/0/0
[*PE2-GigabitEthernet1/0/0] ip binding vpn-instance vpn1
[*PE2-GigabitEthernet1/0/0] ip address 172.16.2.2 24
[*PE2-GigabitEthernet1/0/0] quit
[*PE2] ospf 100 vpn-instance vpn1
[*PE2-ospf-100] import-route bgp
[*PE2-ospf-100] domain-id 10
[*PE2-ospf-100] area 0
[*PE2-ospf-100-area-0.0.0.0] network 172.16.2.0 0.0.0.255
[*PE2-ospf-100-area-0.0.0.0] quit
[*PE2-ospf-100] quit
[*PE2] bgp 100
[*PE2-bgp] ipv4-family vpn-instance vpn1
[*PE2-bgp-vpn1] import-route direct
[*PE2-bgp-vpn1] import-route ospf 100
[*PE2-bgp-vpn1] quit
[*PE2-bgp] quit
[*PE2] commit
```

After completing the configurations, run the **display ip routing-table vpn-instance** command on a PE. You may find that the route to the remote CE is an OSPF route over the user network rather than the BGP route over the backbone network.

The following example uses the command output on PE1.

```
<PE1> display ip routing-table vpn-instance vpn1
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: vpn1
Destinations : 5 Routes : 5
Destination/Mask Proto Pre Cost Flags NextHop Interface
 10.1.0.0/24 OSPF 10 2 D 172.16.1.1 GigabitEthernet1/0/0
 10.2.1.0/24 OSPF 10 3 D 172.16.1.1 GigabitEthernet1/0/0
 172.16.1.0/24 Direct 0 0 D 172.16.1.2 GigabitEthernet1/0/0
 172.16.1.2/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 172.16.2.0/24 OSPF 10 4 D 172.16.1.1 GigabitEthernet1/0/0
```

#### Step 4 Configure a sham link.



To ensure that VPN traffic is forwarded over the MPLS backbone network, ensure that the cost of the sham link is smaller than that of the OSPF route used to forward the traffic over the user network when configuring the sham link. In most cases, you need to change the cost of the interfaces on the user network to ensure that the cost of the OSPF route used to forward the traffic over the user network is greater than that of the sham link.

# Configure CE1.

```
[~CE1] interface GigabitEthernet2/0/0
[~CE1-GigabitEthernet2/0/0] ospf cost 10
[*CE1-GigabitEthernet2/0/0] quit
[*CE1] commit
```

# Configure CE2.

```
[~CE2] interface GigabitEthernet2/0/0
[~CE2-GigabitEthernet2/0/0] ospf cost 10
[*CE2-GigabitEthernet2/0/0] quit
[*CE2] interface loopback 1
[*CE2-LoopBack1] ip address 8.8.8.8 32
[*CE2-LoopBack1] ospf enable 1 area 0
[*CE2-LoopBack1] quit
[*CE2] commit
```

# Configure PE1.

```
[~PE1] interface loopback 10
[*PE1-LoopBack10] ip binding vpn-instance vpn1
[*PE1-LoopBack10] ip address 5.5.5.5 32
[*PE1-LoopBack10] quit
[*PE1] ospf 100 router-id 11.11.11.11
[*PE1-ospf-100] area 0
[*PE1-ospf-100-area-0.0.0.0] sham-link 5.5.5.5 6.6.6.6 cost 1
[*PE1-ospf-100-area-0.0.0.0] quit
[*PE1-ospf-100] quit
[*PE1] commit
```

# Configure PE2.

```
[~PE2] interface loopback 10
[*PE2-LoopBack10] ip binding vpn-instance vpn1
[*PE2-LoopBack10] ip address 6.6.6.6 32
[*PE2-LoopBack10] quit
[*PE2] ospf 100 router-id 22.22.22.22
[*PE2-ospf-100] area 0
[*PE2-ospf-100-area-0.0.0.0] sham-link 6.6.6.6 5.5.5.5 cost 1
[*PE2-ospf-100-area-0.0.0.0] quit
[*PE2-ospf-100] quit
[*PE2] commit
```

### Step 5 Verify the configuration.

After completing the configurations, run the **display ip routing-table vpn-instance** command again on the PE. You may find that the route to the remote CE becomes the BGP route over the backbone network and a route to the destination IP address of the sham link also exist in the routing table.

The following example uses the command output on PE1.

```
<PE1> display ip routing-table vpn-instance vpn1
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: vpn1
Destinations : 10 Routes : 10
Destination/Mask Proto Pre Cost Flags NextHop Interface
 5.5.5.5/32 Direct 0 0 D 127.0.0.1 LoopBack10
 6.6.6.6/32 IBGP 255 0 RD 3.3.3.9 GigabitEthernet2/0/0
 8.8.8.8/32 IBGP 255 2 RD 3.3.3.9 GigabitEthernet2/0/0
 10.1.1.0/24 OSPF 10 11 D 172.16.1.1 GigabitEthernet1/0/0
 10.2.1.0/24 OSPF 10 12 D 172.16.1.1 GigabitEthernet1/0/0
 172.16.1.0/24 Direct 0 0 D 172.16.1.2 GigabitEthernet1/0/0
 172.16.1.2/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 172.16.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 172.16.2.0/24 IBGP 255 0 RD 3.3.3.9 GigabitEthernet2/0/0
 255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

Run the **display ip routing-table** command on a CE. You may find that the cost of the OSPF route to the remote CE becomes 3 and the next hop is the IP address of the connected PE interface. The next hop indicates that the VPN traffic to the remote end is forwarded over the backbone network.

The following example uses the command output on CE1.

```
<CE1> display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Destinations : 15 Routes : 15
Destination/Mask Proto Pre Cost Flags NextHop Interface
 5.5.5.32 O_ASE 150 1 D 172.16.1.2 GigabitEthernet1/0/0
 6.6.6.32 O_ASE 150 1 D 172.16.1.2 GigabitEthernet1/0/0
 8.8.8.32 OSPF 10 3 D 172.16.1.2 GigabitEthernet1/0/0
 10.1.1.0/24 Direct 0 0 D 10.1.1.1 GigabitEthernet2/0/0
 10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
 10.1.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
 10.2.1.0/24 OSPF 10 11 D 10.1.1.2 GigabitEthernet2/0/0
 127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
 127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
 127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
 172.16.1.0/24 Direct 0 0 D 172.16.1.1 GigabitEthernet1/0/0
 172.16.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 172.16.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 172.16.2.0/24 OSPF 10 3 D 172.16.1.2 GigabitEthernet1/0/0
 255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

#### NOTE

Cost (3) of the OSPF route from CE1 to CE2 = Cost (1) of the link from CE1 to PE1 + Cost (1) of the sham link + Cost (1) of the link from PE2 to CE2

Run the **tracert** command and you may find that the next hop for CE1 to send data to CE2 is the IP address of PE1's GE 1/0/0. The next hop indicates that the VPN traffic to the remote end is forwarded over the backbone network.

```
<CE1> tracert 172.16.2.1
traceroute to 172.16.2.1(172.16.2.1), max hops: 30 ,packet length: 40,press CTRL_C to break
1 172.16.1.2 131 ms 2 ms 1 ms
2 10.10.1.2 475 ms 4 ms 4 ms
3 172.16.2.2 150 ms 3 ms 4 ms
4 172.16.2.1 76 ms 3 ms 5 ms
<CE1> tracert 10.2.1.2
traceroute to 10.2.1.2(10.2.1.2), max hops: 30 ,packet length: 40,press CTRL_C to break
1 10.1.1.2 60 ms 1 ms 1 ms
2 10.2.1.2 12 ms 2 ms 2 ms
```

Run the **display ospf sham-link** command on a PE to check whether the sham link is established.

The following example uses the command output on PE1.

```
<PE1> display ospf sham-link
OSPF Process 100 with Router ID 11.11.11.11
Area NeighborID Source-IP Destination-IP State Cost
0.0.0.0 22.22.22.22 5.5.5.5 6.6.6.6 P-2-P 1
```

Run the **display ospf sham-link area** command. The following command output shows that the OSPF neighbor relationship is in Full state.

```
<PE1> display ospf sham-link area 0
OSPF Process 100 with Router ID 11.11.11.11
Sham-Link: 5.5.5.5 --> 6.6.6.6
NeighborID: 22.22.22.22, State: Full, GR status: Normal
Area: 0.0.0
Cost: 1 , State: P-2-P , Type: Sham
Timers: Hello 10 , Dead 40 , Retransmit 5 , Transmit Delay 1
```

Run the **display ospf routing** command on a CE. The following command output shows that the route to the remote CE is an intra-area route.

```
<CE1> display ospf routing
OSPF Process 1 with Router ID 10.1.1.1
Routing Tables
Routing for Network
Destination Cost Type NextHop AdvRouter Area
8.8.8.32 3 Stub 172.16.1.2 10.2.1.2 0.0.0.0
10.1.0/24 10 Direct 10.1.1.1 10.1.1.1 0.0.0.0
10.2.1.0/24 11 Transit 10.1.1.2 10.1.1.2 0.0.0.0
172.16.1.0/24 1 Direct 172.16.1.1 10.1.1.1 0.0.0.0
172.16.2.0/24 3 Transit 172.16.1.2 10.2.1.2 0.0.0.0
Routing for ASEs
Destination Cost Type Tag NextHop AdvRouter
6.6.6.32 1 Type2 3489661028 172.16.1.2 11.11.11.11
5.5.5.32 1 Type2 3489661028 172.16.1.2 22.22.22.22
Total Nets: 7
Intra Area: 5 Inter Area: 0 ASE: 2 NSSA: 0
```

----End

## Configuration Files

- PE1 configuration file

```

sysname PE1

ip vpn-instance vpn1
ipv4-family
 route-distinguisher 100:1
 apply-label per-instance
 vpn-target 1:1 export-extcommunity
 vpn-target 1:1 import-extcommunity

mpls lsr-id 1.1.1.9

mpls

mpls ldp

interface GigabitEthernet1/0/0
undo shutdown
ip binding vpn-instance vpn1
ip address 172.16.1.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.10.1.1 255.255.255.0
mpls
mpls ldp

interface LoopBack1
ip address 1.1.1.9 255.255.255.255

interface LoopBack10
ip binding vpn-instance vpn1
ip address 5.5.5.5 255.255.255.255

bgp 100
peer 3.3.3.9 as-number 100
peer 3.3.3.9 connect-interface LoopBack1

ipv4-family unicast
undo synchronization
peer 3.3.3.9 enable

ipv4-family vpng4
policy vpn-target
peer 3.3.3.9 enable
#
```

```
ipv4-family vpn-instance vpn1
import-route direct
import-route ospf 100
#
ospf 1
area 0.0.0
network 1.1.1.9 0.0.0.0
network 10.10.1.0 0.0.0.255
#
ospf 100 router-id 11.11.11.11 vpn-instance vpn1
import-route bgp
domain-id 0.0.0.10
area 0.0.0
network 172.16.1.0 0.0.0.255
sham-link 5.5.5.5 6.6.6.6 cost 1
#
return
```

- P configuration file

```
#
sysname P
#
mpls lsr-id 2.2.2.9
#
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.10.1.2 255.255.255.0
mpls
mpls ldp
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.20.1.1 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 2.2.2.9 255.255.255.255
#
ospf 1
area 0.0.0
network 2.2.2.9 0.0.0.0
network 10.10.1.0 0.0.0.255
network 10.20.1.0 0.0.0.255
#
return
```

- PE2 configuration file

```
#
sysname PE2
#
ip vpn-instance vpn1
ipv4-family
route-distinguisher 100:2
apply-label per-instance
vpn-target 1:1 export-extcommunity
vpn-target 1:1 import-extcommunity
#
mpls lsr-id 3.3.3.9
#
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
```

```
ip binding vpn-instance vpn1
ip address 172.16.2.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.20.1.2 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 3.3.3.9 255.255.255.255
#
interface LoopBack10
ip binding vpn-instance vpn1
ip address 6.6.6.6 255.255.255.255
#
bgp 100
peer 1.1.1.9 as-number 100
peer 1.1.1.9 connect-interface LoopBack1
#
ipv4-family unicast
undo synchronization
peer 1.1.1.9 enable
#
ipv4-family vpng4
policy vpn-target
peer 1.1.1.9 enable
#
ipv4-family vpn-instance vpn1
import-route direct
import-route ospf 100
#
ospf 1
area 0.0.0
network 3.3.3.9 0.0.0.0
network 10.20.1.0 0.0.0.255
#
ospf 100 router-id 22.22.22.22 vpn-instance vpn1
import-route bgp
domain-id 0.0.0.10
area 0.0.0
network 172.16.2.0 0.0.0.255
sham-link 6.6.6.6 5.5.5.5 cost 1
#
return
```

- CE1 configuration file

```
#
sysname CE1
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
ospf cost 10
#
ospf 1
area 0.0.0
network 172.16.1.0 0.0.0.255
network 10.1.1.0 0.0.0.255
#
return
```

- CE2 configuration file

```
#
sysname CE2
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.2.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0
ospf cost 10
#
interface LoopBack0
ip address 8.8.8.8 255.255.255.255
ospf enable 1 area 0.0.0.0
#
ospf 1
area 0.0.0.0
network 10.2.1.0 0.0.0.255
network 172.16.2.0 0.0.0.255
#
return
```

- RT0 configuration file

```
#
sysname RT0
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0
#
ospf 1
area 0.0.0.0
network 10.1.1.0 0.0.0.255
network 10.2.1.0 0.0.0.255
#
return
```

## 1.1.5 OSPFv3 Configuration

### 1.1.5.1 OSPFv3 Description

#### 1.1.5.1.1 Introduction to OSPFv3

##### Definition

Open Shortest Path First (OSPF) is a link-state Interior Gateway Protocol (IGP) developed by the Internet Engineering Task Force (IETF).

OSPF version 2 (OSPFv2) is intended for IPv4, and OSPF version 3 (OSPFv3) is intended for IPv6.

- OSPFv3 is short for OSPF version 3.
- OSPFv3 runs over IPv6.
- OSPFv3 is an enhanced version of OSPFv2 but is an independent routing protocol.

##### Purpose

OSPFv3 is an extension of OSPF for support of IPv6.

### 1.1.5.1.2 Understanding OSPFv3

#### OSPFv3 Fundamentals

Running on IPv6, OSPFv3 is an independent routing protocol that is developed on the basis of OSPFv2.

- OSPFv3 and OSPFv2 are the same in terms of the working principles of the Hello packet, state machine, link-state database (LSDB), flooding, and route calculation.
- OSPFv3 packets are encapsulated into IPv6 packets and can be transmitted in unicast or multicast mode.

#### OSPFv3 Packet Types

| Packet Type                              | Function                                                                                                                                                      |
|------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Hello packet                             | Hello packets are sent periodically to discover and maintain OSPFv3 neighbor relationships.                                                                   |
| Database Description (DD) packet         | Such packets contain the summary of the local LSDB and are used for LSDB synchronization between two devices.                                                 |
| Link State Request (LSR) packet          | LSR packets are sent to the neighbor to request the required LSAs.<br>An OSPFv3 device sends LSR packets to its neighbor only after they exchange DD packets. |
| Link State Update (LSU) packet           | LSU packets carry the LSAs required by neighbors.                                                                                                             |
| Link State Acknowledgment (LSAck) packet | LSAck packets acknowledge the receipt of an LSA.                                                                                                              |

#### LSA Types

OSPFv3 encapsulates routing information into LSAs for transmission. [Table 1-52](#) describes LSAs and their functions.

**Table 1-52** LSAs and their functions

| LSA Type            | Description                                                                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Router-LSA (Type 1) | Describes the link status and link cost of a device, is generated by the device for the area in which each OSPFv3 interface resides, and is advertised in the area. |

| LSA Type                       | Description                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Network-LSA (Type 2)           | Describes the local link status. Network-LSAs are generated by a designated router (DR) and advertised in the area to which the DR belongs.                                                                                                                                                                                                                   |
| Inter-Area-Prefix-LSA (Type 3) | Describes routes to a specific network segment in an area. Inter-Area-Prefix-LSAs are generated by an ABR and advertised to related areas.                                                                                                                                                                                                                    |
| Inter-Area-Router-LSA (Type 4) | Describes routes to an Autonomous System Boundary Router (ASBR). Inter-Area-Router-LSAs are generated by an ABR and advertised to all related areas except the area to which the ASBR belongs.                                                                                                                                                                |
| AS-external-LSA (Type 5)       | Originated by ASBRs, and advertised to all areas, excluding stub areas and NSSAs. Each AS-external-LSA describes a route to another AS.                                                                                                                                                                                                                       |
| NSSA LSA (Type 7)              | Describes routes to another AS. It is generated by an ASBR and advertised in NSSAs only.                                                                                                                                                                                                                                                                      |
| Link-LSA (Type 8)              | Describes the link-local address and IPv6 address prefix associated with the link and the link option set in the network LSA. Link-LSAs are transmitted only on the link.                                                                                                                                                                                     |
| Intra-Area-Prefix-LSA (Type 9) | Each device or DR generates one or more intra-area prefix LSAs and transmits it in the local area. <ul style="list-style-type: none"> <li>• Such LSAs generated by a device describe the IPv6 address prefixes associated with router-LSAs.</li> <li>• An LSA generated on a DR describes the IPv6 prefix address associated with the network-LSA.</li> </ul> |
| Intra-Area-TE-LSA (Type 10)    | Each device that supports TE services generates LSAs of this type to advertise its own TE information and link-related TE information within the area to which the device belongs.                                                                                                                                                                            |

| LSA Type                                     | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Grace-LSA (Type 11)                          | During the GR process, each device that supports the GR service advertises this type of LSA to each interface. The LSA is advertised by specified interfaces.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| OSPFv3 Router Information (RI) LSA (Type 12) | <p>LSAs of this type are generated when a device needs to advertise its extension information. LSAs of this type are classified into the following types:</p> <ul style="list-style-type: none"><li>• Area Router Information LSA:<br/>LSAs of this type are transmitted in the area to which the device belongs when TLVs such as SRv6 Capabilities TLV, SR-Algorithm TLV, SRv6 NODE MSD TLV, and Dynamic Hostname TLV need to be advertised.</li><li>• AS Router Information LSA<br/>LSAs of this type are transmitted in all areas when TLVs such as Dynamic Hostname TLV need to be advertised.</li></ul> |
| E-Router-LSA (Type 33)                       | LSAs of this type are generated when a device needs to advertise link extension TLVs, such as the End.X SID Sub-TLV and Local Interface IPv6 Address Sub-TLV. Such LSAs are transmitted in the area to which the device belongs.                                                                                                                                                                                                                                                                                                                                                                              |
| E-AS-External-LSA (Type 37)                  | Describes extended TLV information associated with AS-external routes and are generated by an ASBR and advertised to all areas.                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| SRv6 Locator LSA (Type 42)                   | Each SRv6-capable device generates this type of LSA for each area when advertising SRv6 locators. These LSAs are transmitted in each area.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

## Router Types

Figure 1-141 Router types

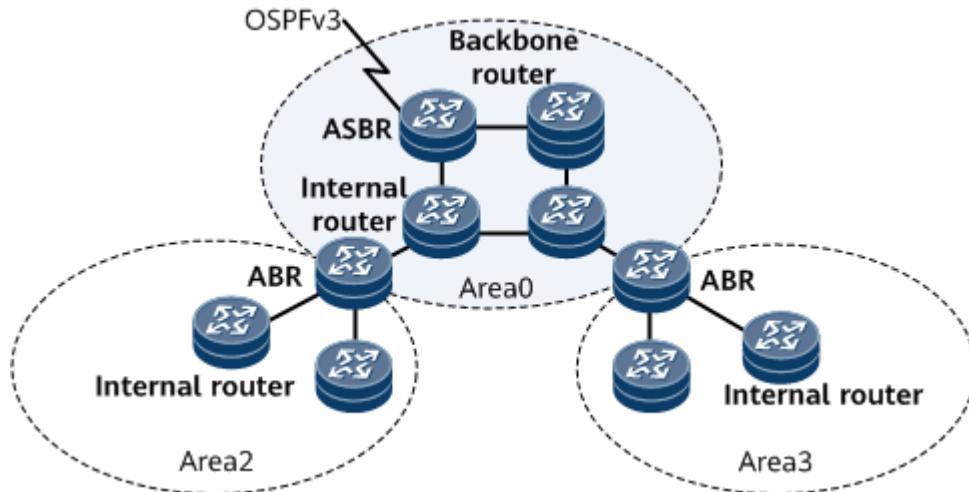


Table 1-53 Router types and descriptions

| Router Type               | Description                                                                                                                                                                                                                                       |
|---------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Internal router           | All interfaces on an internal router belong to the same OSPFv3 area.                                                                                                                                                                              |
| Area border router (ABR)  | An ABR belongs to two or more areas, one of which must be the backbone area.<br>An ABR is used to connect the backbone area and non-backbone areas. It can be physically or logically connected to the backbone area.                             |
| Backbone router           | At least one interface on a backbone router belongs to the backbone area.<br>Internal routers in Area 0 and all ABRs are backbone routers.                                                                                                        |
| AS boundary router (ASBR) | An ASBR exchanges routing information with other ASs.<br>An ASBR does not necessarily reside on the border of an AS. It can be an internal router or an ABR. An OSPFv3 device that has imported external routing information will become an ASBR. |

## OSPFv3 Route Types

Inter-area routes and intra-area routes describe the network structure of an AS. External routes describe how to select a route to the destination outside an AS. OSPFv3 classifies the imported AS external routes into Type 1 routes and Type 2 routes.

Table 1-54 lists route types in descending order of priority.

**Table 1-54** Types of OSPFv3 routes

| Route Type            | Description                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Intra Area            | Indicates routes within an area.                                                                                                                                                                                                                                                                                                                             |
| Inter Area            | Indicates routes between areas.                                                                                                                                                                                                                                                                                                                              |
| Type 1 external route | Such routes offer higher reliability, and their costs are approximately the same as those of AS internal routes and are comparable with the costs of routes generated by OSPFv3.<br><br>Cost of a Type 1 external route = Cost of the route from a local router to an ASBR + Cost of the route from the ASBR to the destination of the Type 1 external route |
| Type 2 external route | Such routes have low reliability. Therefore, OSPFv3 considers that the cost of the route from an ASBR to the destination outside the AS is much greater than the cost of any internal route to the ASBR.<br><br>Cost of a Type 2 external route = Cost of the route from the ASBR to the destination of the Type 2 external route                            |

## Area

When a large number of routers run OSPFv3, LSDBs become very large and require a large amount of storage space. Large LSDBs also complicate shortest path first (SPF) computation and may overload routers. As the network scale expands, the probability of network topology changes increases, which causes the network to continuously change. In such cases, large numbers of OSPFv3 packets are transmitted on the network, leading to a decrease in bandwidth utilization efficiency. Each change in the network topology causes all routers on the network to recalculate routes.

OSPFv3 resolves this problem by partitioning an AS into different areas, improving network utilization. An area is regarded as a logical group, and each group is identified by an area ID. A router, not a link, resides at the border of an area. A network segment or link can belong only to one area. An area must be specified for each OSPFv3 interface.

OSPFv3 areas include common areas, stub areas, and NSSAs. **Table 1-55** describes these in more details.

**Table 1-55** Area types

| Area Type   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | Notes                                                                                                                                                                                                                               |
|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Common area | <p>OSPFv3 areas are common areas by default. Common areas include standard areas and backbone areas.</p> <ul style="list-style-type: none"> <li>• Standard area: transmits intra-area, inter-area, and external routes.</li> <li>• Backbone area: connects to all other OSPFv3 areas and transmits inter-area routes. The backbone area is represented by area 0. Routes between non-backbone areas must be forwarded through the backbone area.</li> </ul>                                                                                                                                                                                                                                                                              | <ul style="list-style-type: none"> <li>• In the backbone area, all devices must be connected.</li> <li>• All non-backbone areas must be connected to the backbone area.</li> </ul>                                                  |
| Stub area   | <p>A stub area is a non-backbone area with only one ABR and generally resides at the border of an AS. The ABR in a stub area does not transmit received AS external routes, which significantly decreases the number of entries in the routing table on the ABR (router) and the amount of routing information to be transmitted. To ensure the reachability of AS external routes, the ABR in the stub area generates a default route and advertises the route to non-ABRs in the stub area.</p> <p>A totally stubby area allows only intra-area routes and ABR-advertised Type 3 default routes to be advertised within the area. The totally stubby area does not allow AS external routes or inter-area routes to be advertised.</p> | <ul style="list-style-type: none"> <li>• The backbone area cannot be configured as a stub area.</li> <li>• An ASBR cannot exist in a stub area. Therefore, AS external routes cannot be advertised within the stub area.</li> </ul> |
| NSSA        | <p>An NSSA is similar to a stub area. An NSSA does not advertise Type 5 LSAs but can import AS external routes. ASBRs in an NSSA generate Type 7 LSAs to carry the information about the AS external routes. The Type 7 LSAs are advertised only within the NSSA. When the Type 7 LSAs reach an ABR in the NSSA, the ABR translates the Type 7 LSAs into Type 5 LSAs and floods them to the entire OSPFv3 domain.</p> <p>A totally NSSA area allows only intra-area routes to be advertised within the area.</p>                                                                                                                                                                                                                         | <ul style="list-style-type: none"> <li>• ABRs in an NSSA advertise Type 3 LSAs carrying a default route within the NSSA. All inter-area routes are advertised by ABRs.</li> </ul>                                                   |

## Network Types Supported by OSPFv3

OSPFv3 classifies networks into the following types (listed in **Table 1-56**) based on link layer protocols.

**Table 1-56** Types of OSPFv3 networks

| Network Type                         | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Broadcast                            | <p>OSPFv3 considers networks with Ethernet or Fiber Distributed Data Interface (FDDI) as the link layer protocol as broadcast networks by default.</p> <p>On this type of network:</p> <ul style="list-style-type: none"><li>• Hello packets, LSU packets, and LSAck packets are usually transmitted in multicast mode. FF02::5 is an IPv6 multicast address reserved for an OSPFv3 device. FF02::6 is an IPv6 multicast address reserved for an OSPFv3 DR or backup designated router (BDR).</li><li>• DD and LSR packets are transmitted in unicast mode.</li></ul> |
| Non-broadcast Multiple Access (NBMA) | <p>OSPFv3 considers networks with X.25 as the link layer protocol as NBMA networks by default.</p> <p>On an NBMA network, protocol packets, such as Hello packets, DD packets, LSR packets, LSU packets, and LSAck packets are sent in unicast mode.</p>                                                                                                                                                                                                                                                                                                              |
| Point-to-Multipoint (P2MP)           | <p>No network is a P2MP network by default, no matter what type of link layer protocol is used on the network. A non-fully meshed NBMA network can be changed to a P2MP network.</p> <p>On this type of network:</p> <ul style="list-style-type: none"><li>• Hello packets are transmitted in multicast mode using the multicast address FF02::5.</li><li>• Other types of protocol packets, such as DD packets, LSR packets, LSU packets, and LSAck packets are sent in unicast mode.</li></ul>                                                                      |
| Point-to-point (P2P)                 | <p>If the link layer protocol is PPP, LAPB, OSPFv3 defaults the network type to P2P.</p> <p>On such a type of network, protocol packets, such as Hello packets, DD packets, LSR packets, LSU packets, and LSAck packets are sent in multicast mode using the multicast address FF02::5.</p>                                                                                                                                                                                                                                                                           |

## Stub Area

Stub areas are specific areas where ABRs do not flood received AS external routes. In stub areas, routers maintain fewer routing entries and less routing information than the routers in other areas.

Configuring a stub area is optional. Not every area can be configured as a stub area, because a stub area is usually a non-backbone area with only one ABR and is located at the AS border.

To ensure the reachability of the routes to destinations outside an AS, the ABR in the stub area generates a default route and advertises the route to the non-ABRs in the same stub area.

Note the following points when configuring a stub area:

- The backbone area cannot be configured as a stub area.
- If an area needs to be configured as a stub area, all the devices in the area must be configured as stub devices using the **stub** command.
- An ASBR cannot exist in a stub area. Therefore, AS external routes cannot be advertised within the stub area.

## NSSA

Stub areas cannot import or transmit external routes, which prevents a large number of external routes from consuming the bandwidth and storage resources of routers in the stub areas. If you need to import external routes to an area and prevent these routes from consuming resources, configure the area as a not-so-stubby area (NSSA).

As a new type of area in OSPFv3, NSSAs resemble stub areas in many ways. Different from stub areas, NSSAs can import AS external routes and advertise them within the entire OSPFv3 AS, without learning external routes from other areas.

To advertise external routes imported by an NSSA to other areas on the OSPFv3 network, a translator must translate Type 7 LSAs into Type 5 LSAs.

### NOTE

- The propagate bit (P-bit) is used to notify a translator whether Type 7 LSAs need to be translated.
- By default, the translator is the ABR with the largest router ID in the NSSA.
- The P-bit is not set for Type 7 LSAs generated by an ABR.

## OSPFv3 Route Summarization

Routes with the same IPv6 prefix can be summarized into one route. On a large-scale OSPFv3 network, route lookup may slow down because of the large size of the routing table. To reduce the routing table size and simplify management, configure route summarization. With route summarization, if a link connected to a device within an IPv6 address range that has been summarized alternates between Up and Down, the link status change is not advertised to the devices beyond the IPv6 address range. This prevents route flapping and improves network stability.

OSPFv3 route summarization is classified as follows:

- Route summarization on an ABR

An ABR can summarize routes with the same prefix into one route and advertise the summarized route to other areas.

When an ABR transmits routing information to other areas, it generates Type 3 LSAs based on IPv6 address prefixes. If contiguous IPv6 address prefixes exist in this area and summarization is enabled on the ABR, these IPv6 address prefixes are summarized into one address prefix so that the ABR sends only one summary LSA. The LSAs of the specified network segments are not advertised.

- Route summarization on an ASBR

An ASBR can summarize imported routes with the same prefix into one route and then advertise the summarized route to other areas.

With route summarization, an ASBR summarizes imported Type 5 LSAs within the summarized address range. After route summarization, the ASBR does not generate a separate Type 5 LSA for each specific prefix within the configured range. Instead, the ASBR generates a Type 5 LSA for only the summarized prefix. In an NSSA, an ASBR summarizes multiple imported Type 7 LSAs within the summary address range into one Type 7 LSA.

## OSPFv3 Multi-process

OSPFv3 supports multi-process. Multiple OSPFv3 processes can independently run on the same router. Route exchange between different OSPFv3 processes is similar to that between different routing protocols.

## OSPFv3 and OSPFv2 Comparison

### Similarities:

- Network types and interface types
- Interface and neighbor state machines
- LSDB
- Flooding mechanism
- Five types of packets: Hello, DD, LSR, LSU, and LSAck packets
- Route calculation

### Differences:

- In OSPFv3, only LSAs in LSU packets contain address information.
- OSPFv3 is based on links rather than network segments.  
OSPFv3 runs over IPv6, which is based on links instead of network segments. As such, the interfaces on which OSPFv3 is to be configured must be on the same link, rather than on the same network segment. In addition, the interfaces can establish OSPFv3 connections without IPv6 global addresses.
- OSPFv3 does not depend on IP addresses.  
OSPFv3 separates topology calculation from IP addresses. OSPFv3 can calculate the OSPFv3 topology without knowing IPv6 global addresses, which are used only for virtual link interfaces and packet forwarding.
- OSPFv3 packets and the LSA format change.
  - OSPFv3 packets do not contain IP addresses.
  - OSPFv3 router-LSAs and network-LSAs do not contain IP addresses, which are advertised by link-LSAs and intra-area-prefix-LSAs.

- In OSPFv3, router IDs, area IDs, and LSA link state IDs no longer indicate IP addresses, but the IPv4 address format is still reserved.
  - Neighbors are identified by router IDs instead of IP addresses on broadcast, NBMA, or P2MP networks.
- Information about the flooding scope is added to OSPFv3 LSAs.  
Information about the flooding scope is added to the LSA Type field of OSPFv3 LSAs. Therefore, OSPFv3 routers can process LSAs of unidentified types.
    - OSPFv3 can store or flood unidentified packets, whereas OSPF discards unidentified packets.
    - OSPFv3 can flood unknown LSAs with the U bit set to 1. The LSA flooding scope is specified by the OSPFv3 LSA itself.
- For example, DeviceA and DeviceB can identify LSAs of a certain type. They are connected through DeviceC, but DeviceC cannot identify LSAs of this type. In this manner, when DeviceA floods such LSAs, DeviceC floods them to DeviceB without identifying them. After receiving the LSAs, DeviceB processes them.
- If OSPF is run, the unidentified LSAs are discarded, unable to reach DeviceB.
- OSPFv3 supports multi-process on a link.  
In OSPFv2, one physical interface can be bound to only one multi-instance. In OSPFv3, one physical interface can be bound to multiple instances, which are identified by different instance IDs. Each of these OSPFv3 instances can establish a neighbor relationship with the remote end of the same physical link and transmit packets. These instances do not interfere with each other. As such, link resources can be fully shared among these instances.
  - OSPFv3 uses IPv6 link-local addresses.  
IPv6 implements neighbor discovery, automatic configuration, and other functions based on link-local addresses. Devices running IPv6 do not forward IPv6 packets whose destination address is a link-local address. Those packets can only be exchanged on the same link. The unicast link-local address starts from FE80/10.  
As a routing protocol running over IPv6, OSPFv3 uses link-local addresses to maintain neighbor relationships and synchronize LSA databases. All OSPFv3 interfaces, except virtual link interfaces, use link-local addresses as the source address and the next hop address to transmit OSPFv3 packets.
- The advantages are as follows:
- OSPFv3 can calculate topologies without global IPv6 addresses, separating topologies from addresses.
  - The packets flooded on a link are not transmitted to other links. This minimizes unnecessary flooding and thereby saves bandwidth resources.
- OSPFv3 supports two new LSAs.
    - Link LSA: A device floods a link LSA on the link where it resides to advertise its link-local address and the configured global IPv6 address.
    - Intra-area prefix LSA: A device advertises an intra-area prefix LSA in the local area to inform the other devices in the area or the network (either a broadcast network or an NBMA network) of its IPv6 global address.

OSPFv3 identifies neighbors based on router IDs only.

On broadcast, NBMA, and P2MP networks, OSPF identifies neighbors based on the IPv4 addresses of interfaces.

## OSPFv3 Packet Format

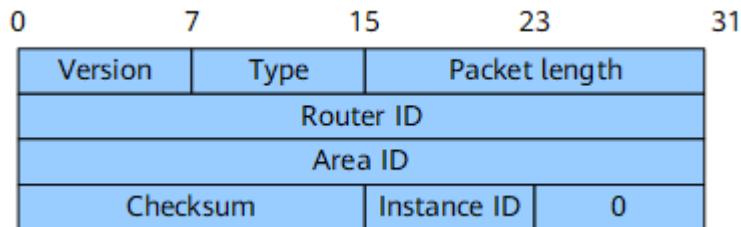
Open Shortest Path First (OSPF) for IPv6 packets are encapsulated into IPv6 packets. The OSPFv3 protocol number is 89. OSPFv3 packets are classified into the following types:

- **Hello packet**
- **Database Description (DD) packet**
- **Link State Request (LSR) packet**
- **Link State Update (LSU) packet**
- **Link State Acknowledgment (LSAck) packet**

## Packet Header Format

The five types of OSPFv3 packets have the same packet header format. The length of an OSPFv3 packet header is 16 bytes. [Figure 1-142](#) shows an OSPFv3 packet header.

**Figure 1-142** OSPFv3 packet header



**Table 1-57** Packet header fields

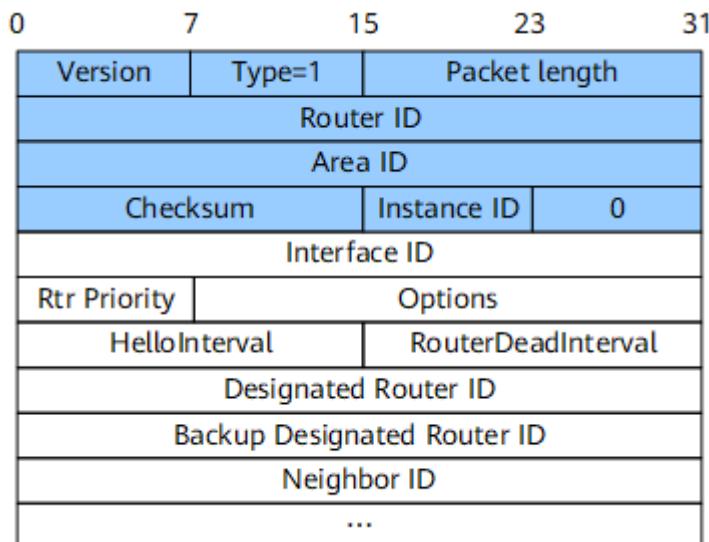
| Field         | Length  | Description                                                                                                                                                                                                       |
|---------------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Version       | 8 bits  | OSPF version number. For OSPFv3, the value is 3.                                                                                                                                                                  |
| Type          | 8 bits  | OSPFv3 packet type. The values are as follows: <ul style="list-style-type: none"><li>• 1: Hello packet</li><li>• 2: DD packet</li><li>• 3: LSR packet</li><li>• 4: LSU packet</li><li>• 5: LSAck packet</li></ul> |
| Packet length | 16 bits | Length of the OSPFv3 packet containing the packet header, in bytes.                                                                                                                                               |
| Router ID     | 32 bits | ID of the router that sends the OSPFv3 packet.                                                                                                                                                                    |
| Area ID       | 32 bits | ID of the area to which the router that sends the OSPFv3 packet belongs.                                                                                                                                          |

| Field       | Length  | Description                                                                   |
|-------------|---------|-------------------------------------------------------------------------------|
| Checksum    | 16 bits | Checksum of the OSPFv3 packet that does not contain the Authentication field. |
| Instance ID | 8 bits  | ID of an OSPFv3 instance.                                                     |
| 0           | 8 bits  | Reserved fields.                                                              |

## Hello Packet

Hello packets are commonly used packets, which are periodically sent on OSPFv3 interfaces to establish and maintain neighbor relationships. A Hello packet includes information about the designated router (DR), backup designated router (BDR), timers, and known neighbors. **Figure 1-143** shows the format of a Hello packet.

**Figure 1-143** Format of a Hello packet



**Table 1-58** Hello packet fields

| Field        | Length  | Description                                                                                                                                                            |
|--------------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Interface ID | 32 bits | ID of the interface that sends the Hello packets.                                                                                                                      |
| Rtr Priority | 8 bits  | DR priority. The default value is 1.<br><b>NOTE</b><br>If the DR priority of a router interface is set to 0, the interface cannot participate in a DR or BDR election. |

| Field                       | Length  | Description                                                                                                                                                                                                                                                                   |
|-----------------------------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Options                     | 24 bits | The values are as follows: <ul style="list-style-type: none"><li>• E: Type 5 link state advertisements (LSAs) are flooded.</li><li>• MC: IP multicast packets are forwarded.</li><li>• N/P: Type 7 LSAs are processed.</li><li>• DC: On-demand links are processed.</li></ul> |
| HelloInterval               | 16 bits | Interval at which Hello packets are sent.                                                                                                                                                                                                                                     |
| RouterDeadInterval          | 16 bits | Dead interval. If a router does not receive any Hello packets from its neighbors within a specified dead interval, the neighbors are considered Down.                                                                                                                         |
| Designated Router ID        | 32 bits | Router ID of the DR.                                                                                                                                                                                                                                                          |
| Backup Designated Router ID | 32 bits | Router ID of the BDR.                                                                                                                                                                                                                                                         |
| Neighbor ID                 | 32 bits | Router ID of the neighbor.                                                                                                                                                                                                                                                    |

**Table 1-59** lists the address types, interval types, and default intervals used when Hello packets are transmitted on different networks.

**Table 1-59** Hello packet characteristics for various network types

| Network Type                         | Address Type      | Interval Type                                                                                                                                                                                                     | Default Interval                                             |
|--------------------------------------|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------|
| Broadcast                            | Multicast address | HelloInterval                                                                                                                                                                                                     | 10 seconds                                                   |
| Non-broadcast multiple access (NBMA) | Unicast address   | <ul style="list-style-type: none"><li>• HelloInterval for the DR, BDR, and router that can become a DR</li><li>• PollInterval for the case when neighbors become Down and HelloInterval for other cases</li></ul> | 30 seconds for HelloInterval<br>120 seconds for PollInterval |
| Point-to-point (P2P)                 | Multicast address | HelloInterval                                                                                                                                                                                                     | 10 seconds                                                   |

| Network Type               | Address Type      | Interval Type | Default Interval |
|----------------------------|-------------------|---------------|------------------|
| Point-to-multipoint (P2MP) | Multicast address | HelloInterval | 30 seconds       |

#### NOTE

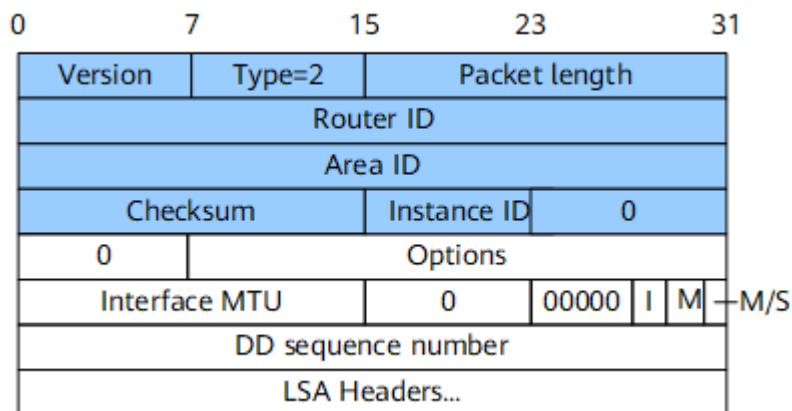
To establish neighbor relationships between routers on the same network segment, you must set the same HelloInterval, PollInterval, and RouterDeadInterval values for the routers. PollInterval applies only to NBMA networks.

## DD Packet

During an adjacency initialization, two routers use DD packets to describe their own link state databases (LSDBs) for LSDB synchronization. A DD packet contains the header of each LSA in an LSDB. An LSA header uniquely identifies an LSA. The LSA header occupies only a small portion of the LSA, which reduces the amount of traffic transmitted between routers. A neighbor can use the LSA header to check whether it already has the LSA. When two routers exchange DD packets, one functions as the master and the other functions as the slave. The master defines a start sequence number. The master increases the sequence number by one each time it sends a DD packet. After the slave receives a DD packet, it uses the sequence number carried in the DD packet for acknowledgment.

[Figure 1-144](#) shows the format of a DD packet.

[Figure 1-144](#) Format of a DD packet



**Table 1-60** DD packet fields

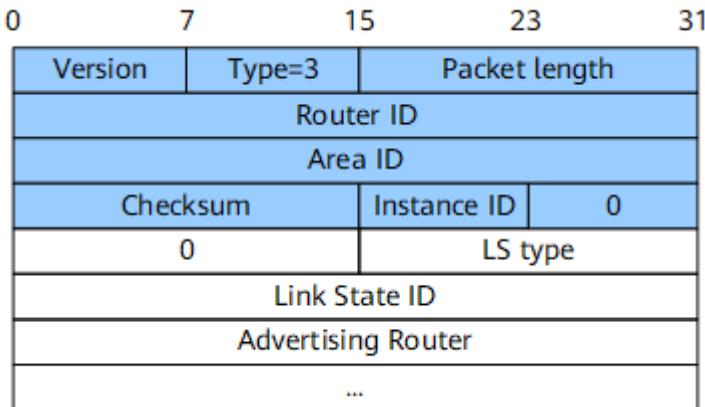
| Field              | Length  | Description                                                                                                                                                                                                                                       |
|--------------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Options            | 24 bits | The values are as follows: <ul style="list-style-type: none"><li>● E: Type 5 LSAs are flooded.</li><li>● MC: IP multicast packets are forwarded.</li><li>● N/P: Type 7 LSAs are processed.</li><li>● DC: On-demand links are processed.</li></ul> |
| Interface MTU      | 16 bits | Maximum length of the DD packet sent by the interface with packet fragmentation disabled.                                                                                                                                                         |
| I                  | 1 bit   | If the DD packet is the first packet among multiple consecutive DD packets sent by a router, this field is set to 1. In other cases, this field is set to 0.                                                                                      |
| M (More)           | 1 bit   | If the DD packet is the last packet among multiple consecutive DD packets sent by a router, this field is set to 0. In other cases, this field is set to 1.                                                                                       |
| M/S (Master/Slave) | 1 bit   | When two routers exchange DD packets, they negotiate a master/slave relationship. The router with a larger router ID becomes the master. If this field is set to 1, the DD packet is sent by the master.                                          |
| DD sequence number | 32 bits | Sequence number of the DD packet. The master and slave use the sequence number to ensure that DD packets are correctly transmitted.                                                                                                               |
| LSA Headers        | -       | LSA header information included in the DD packet.                                                                                                                                                                                                 |

## LSR Packet

After two routers exchange DD packets, they send LSR packets to request each other's LSAs. The LSR packets contain the summaries of the requested LSAs.

[Figure 1-145](#) shows the format of an LSR packet.

[Figure 1-145](#) Format of an LSR packet



**Table 1-61** LSR packet fields

| Field              | Length  | Description                                                           |
|--------------------|---------|-----------------------------------------------------------------------|
| LS type            | 16 bits | Type of the LSA                                                       |
| Link State ID      | 32 bits | This field together with the LS type field describes an LSA in an AS. |
| Advertising Router | 32 bits | Router ID of the router that generates the LSA.                       |

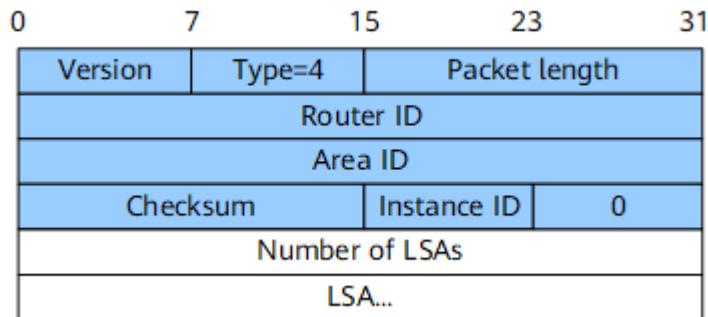
 **NOTE**

The LS type, Link State ID, and Advertising Router fields can uniquely identify an LSA. If two LSAs have the same LS type, Link State ID, and Advertising Router fields, a router uses the LS sequence number, LS checksum, and LS age fields to obtain a required LSA.

## LSU Packet

A router uses an LSU packet to transmit LSAs requested by its neighbors or to flood its own updated LSAs. The LSU packet contains a set of LSAs. For multicast and broadcast networks, LSU packets are multicast to flood LSAs. To ensure reliable LSA flooding, a router uses an LSAck packet to acknowledge the LSAs contained in an LSU packet that is received from a neighbor. If an LSA fails to be acknowledged, the router retransmits the LSA to the neighbor. [Figure 1-146](#) shows the format of an LSU packet.

[Figure 1-146](#) Format of an LSU packet



**Table 1-62** LSU packet field

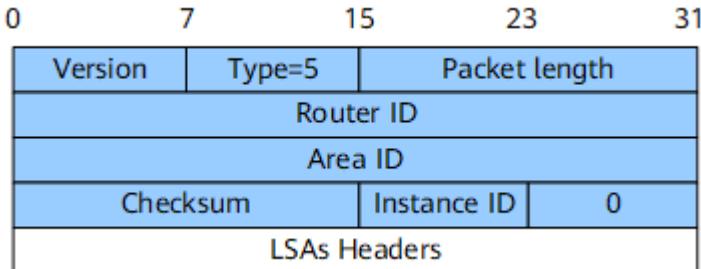
| Field          | Length  | Description                                |
|----------------|---------|--------------------------------------------|
| Number of LSAs | 32 bits | Number of LSAs contained in the LSU packet |

## LSAck Packet

A router uses an LSAck packet to acknowledge the LSAs contained in a received LSU packet. The LSAs can be acknowledged using LSA headers. LSAck packets can

be transmitted over different links in unicast or multicast mode. [Figure 1-147](#) shows the format of an LSAck packet.

**Figure 1-147** Format of an LSAck packet



**Table 1-63** LSAck packet field

| Field        | Length                                                         | Description                               |
|--------------|----------------------------------------------------------------|-------------------------------------------|
| LSAs Headers | Determined by the header length of the LSA to be acknowledged. | This field is used to acknowledge an LSA. |

## OSPFv3 LSA Format

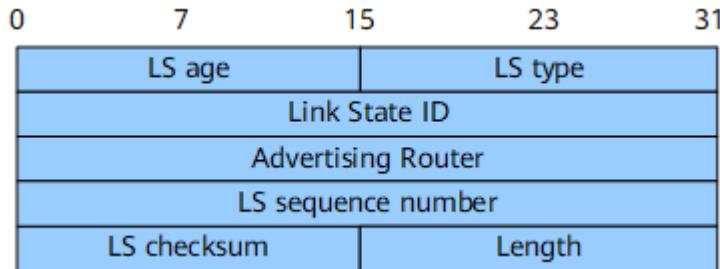
Each router in an autonomous system (AS) generates one or more types of link state advertisements (LSAs), depending on the router's type. Multiple LSAs form a link state database (LSDB). OSPFv3 encapsulates routing information into LSAs for transmission. Commonly used LSAs include:

- [Router-LSA \(Type 1\)](#)
- [Network-LSA \(Type 2\)](#)
- [Inter-Area-Prefix-LSA \(Type 3\)](#)
- [Inter-Area-Router-LSA \(Type 4\)](#)
- [AS-external-LSA \(Type 5\)](#)
- [NSSA LSA \(Type 7\)](#)
- [Link-LSA \(Type 8\)](#)
- [Intra-Area-Prefix-LSA \(Type 9\)](#)

## LSA Header Format

All LSAs have the same header. [Figure 1-148](#) shows an LSA header.

**Figure 1-148 LSA header**



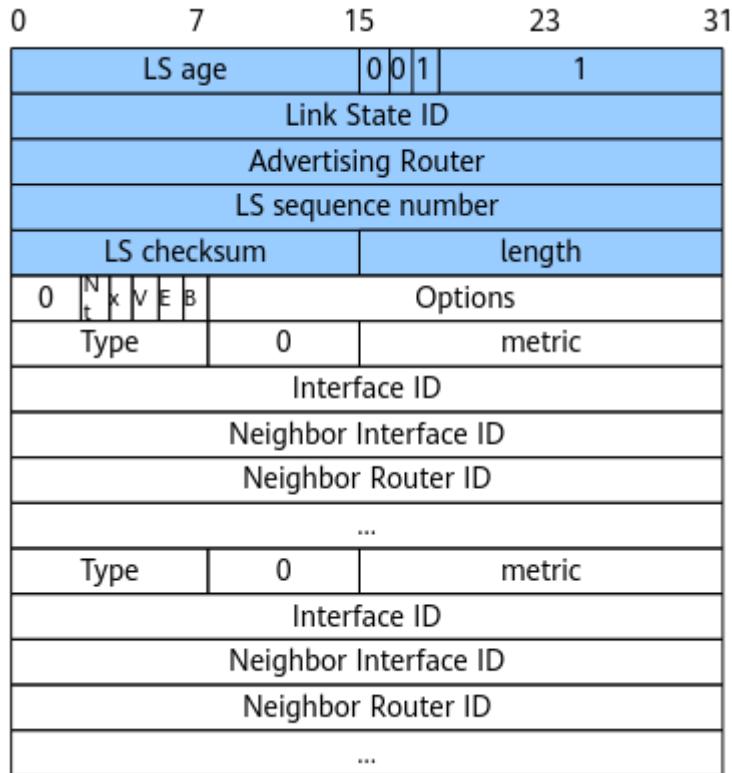
**Table 1-64 LSA header fields**

| Field              | Length  | Description                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LS age             | 16 bits | Time that elapses after the LSA is generated, in seconds. The value of this field continually increases regardless of whether the LSA is transmitted over a link or saved in an LSDB.                                                                                                                                                                                           |
| LS type            | 16 bits | Type of the LSA. The values are as follows: <ul style="list-style-type: none"> <li>• Type 1: Router-LSA.</li> <li>• Type 2: Network-LSA.</li> <li>• Type 3: Inter-Area-Prefix-LSA.</li> <li>• Type 4: Inter-Area-Router-LSA.</li> <li>• Type 5: AS-external-LSA.</li> <li>• Type 7: NSSA-LSA.</li> <li>• Type 8: Link-LSA.</li> <li>• Type 9: Intra-Area-Prefix-LSA.</li> </ul> |
| Link State ID      | 32 bits | This field together with the LS type field describes an LSA in an area.                                                                                                                                                                                                                                                                                                         |
| Advertising Router | 32 bits | Router ID of the router that generates the LSA.                                                                                                                                                                                                                                                                                                                                 |
| LS sequence number | 32 bits | Sequence number of the LSA. Routers can use this field to identify the latest LSA.                                                                                                                                                                                                                                                                                              |
| LS checksum        | 16 bits | Checksum of all fields except the LS age field.                                                                                                                                                                                                                                                                                                                                 |
| Length             | 16 bits | Length of the LSA including the LSA header, in bytes.                                                                                                                                                                                                                                                                                                                           |

## Router-LSA

A router-LSA (Type 1) describes the link status and cost of a router. Router-LSAs are generated by a router and advertised within the area to which the router belongs. [Figure 1-149](#) shows the format of a router-LSA.

**Figure 1-149** Format of a router-LSA



**Table 1-65** Router-LSA fields

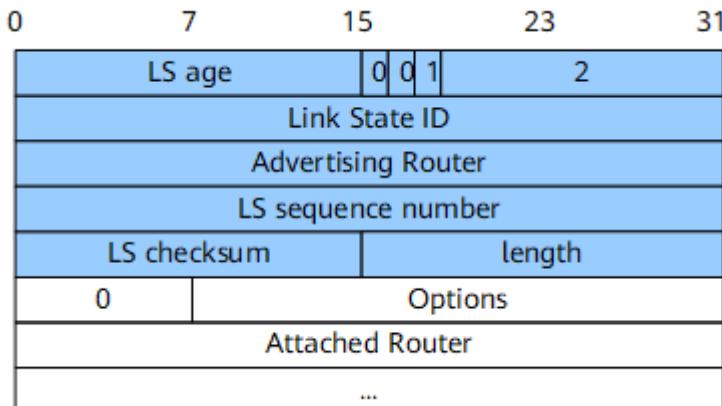
| Field                 | Length  | Description                                                                                                                                                                                                                        |
|-----------------------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Nt (NSSA translation) | 1 bit   | If the router that generates the LSA is an NSSA border router, this field is set to 1. In other cases, this field is set to 0. When this field is set to 1, the router unconditionally translates NSSA-LSAs into AS-external-LSAs. |
| x                     | 1 bit   | This field is deprecated.                                                                                                                                                                                                          |
| V (Virtual Link)      | 1 bit   | If the router that generates the LSA is located at one end of a virtual link, this field is set to 1. In other cases, this field is set to 0.                                                                                      |
| E (External)          | 1 bit   | If the router that generates the LSA is an autonomous system boundary router (ASBR), this field is set to 1. In other cases, this field is set to 0.                                                                               |
| B (Border)            | 1 bit   | If the router that generates the LSA is an area border router (ABR), this field is set to 1. In other cases, this field is set to 0.                                                                                               |
| Options               | 24 bits | The optional capabilities supported by the router.                                                                                                                                                                                 |

| Field                 | Length  | Description                                                                                                                                                                                                                                                              |
|-----------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Type                  | 8 bits  | Type of the router link. The values are as follows: <ul style="list-style-type: none"> <li>• 1: Connected to another router in point-to-point (P2P) mode.</li> <li>• 2: Connected to a transport network.</li> <li>• 3: Reserved.</li> <li>• 4: Virtual link.</li> </ul> |
| metric                | 16 bits | Cost of the link.                                                                                                                                                                                                                                                        |
| Interface ID          | 32 bits | The Interface ID assigned to the interface.                                                                                                                                                                                                                              |
| Neighbor Interface ID | 32 bits | Neighbor's interface ID. <ul style="list-style-type: none"> <li>• For transit links, the value is the interface ID of the DR.</li> <li>• For links of other types, the value is the interface ID of the neighboring device.</li> </ul>                                   |
| Neighbor Router ID    | 32 bits | Router ID of the neighbor. <ul style="list-style-type: none"> <li>• For transit links, the value is the router ID of the DR.</li> <li>• For links of other types, the value is the router ID of the neighboring device.</li> </ul>                                       |

## Network-LSA

A network-LSA (Type 2) records the router IDs of all routers on the local network segment. Network-LSAs are generated by a DR on a broadcast or non-broadcast multiple access (NBMA) network and advertised within the area to which the DR belongs. [Figure 1-150](#) shows the format of a network-LSA.

**Figure 1-150** Format of a network-LSA



**Table 1-66** Network-LSA fields

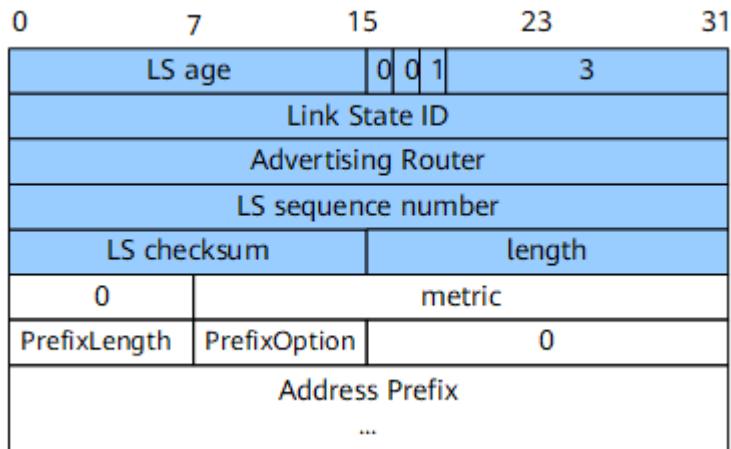
| Field           | Length  | Description                                                                      |
|-----------------|---------|----------------------------------------------------------------------------------|
| Options         | 24 bits | The optional capabilities supported by the router.                               |
| Attached Router | 32 bits | Router IDs of all routers on the same network, including the router ID of the DR |

## Inter-Area-Prefix-LSA

An inter-area-prefix-LSA (Type 3) describes routes on a network segment in an area. It is generated by the ABR. The routes are advertised to other areas.

[Figure 1-151](#) shows the format of an inter-area-prefix-LSA.

**Figure 1-151** Format of an inter-area-prefix-LSA



**Table 1-67** Inter-area-prefix-LSA fields

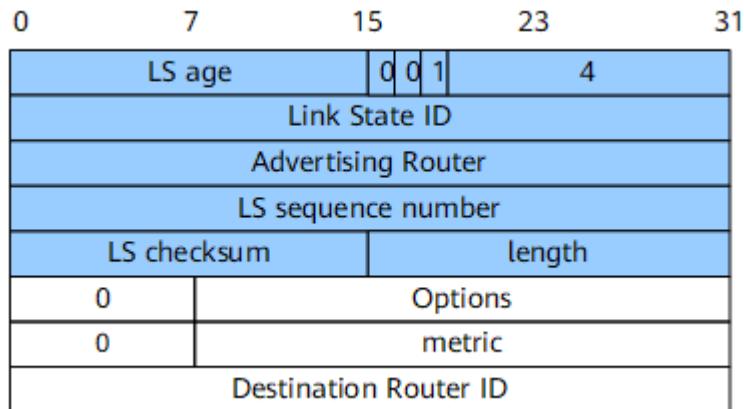
| Field          | Length  | Description                                                            |
|----------------|---------|------------------------------------------------------------------------|
| PrefixLength   | 8 bits  | Length of the prefix.                                                  |
| PrefixOption   | 8 bits  | Prefix-related capability option, indicating the length in the packet. |
| Address Prefix | 32 bits | Address prefix.                                                        |

## Inter-Area-Router-LSA

An inter-area-router-LSA (Type 4) describes routes to the ASBR in other areas. It is generated by the ABR. The routes are advertised to all related areas except the area that the ASBR belongs to.

[Figure 1-152](#) shows the format of an inter-area-router-LSA.

**Figure 1-152** Format of an inter-area-router-LSA



**Table 1-68** Inter-area-router-LSA fields

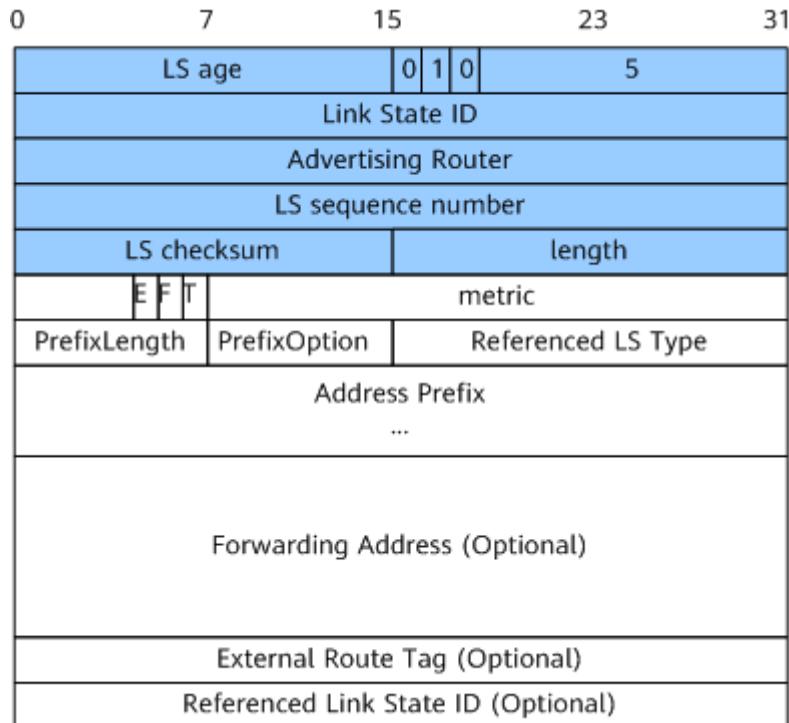
| Field                 | Length  | Description                                       |
|-----------------------|---------|---------------------------------------------------|
| Destination Router ID | 32 bits | The router ID of the router described by the LSA. |

## AS-External-LSA

An AS-external-LSA describes a route to a destination outside the AS and is generated by an ASBR.

**Figure 1-153** shows the format of an AS-external-LSA.

**Figure 1-153** Format of an AS-external-LSA



**Table 1-69 AS-external-LSA fields**

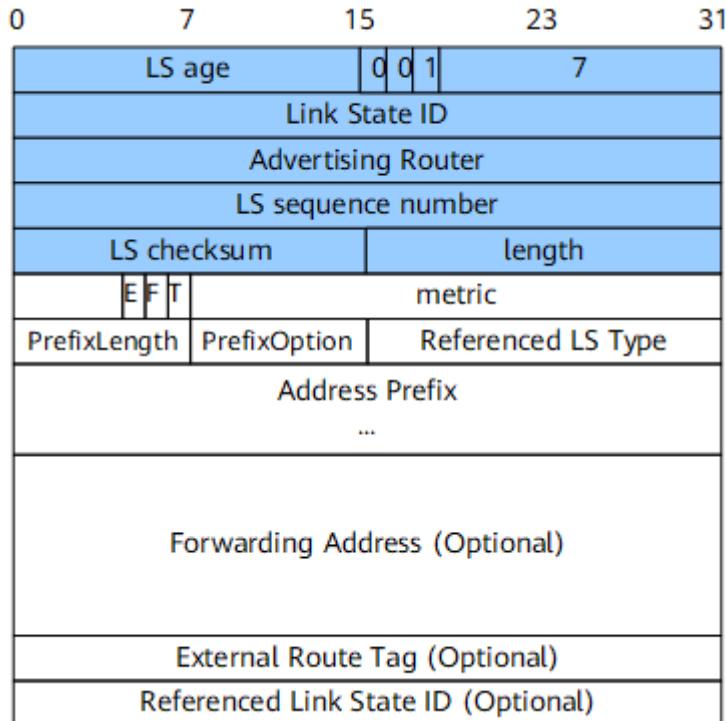
| Field                    | Length   | Description                                                                                                                                                                                                                                                       |
|--------------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| E                        | 1 bit    | Type of external metric. <ul style="list-style-type: none"><li>● If this field is 1, the specified metric is a Type 2 external metric.</li><li>● If this field is 0, the specified metric is a Type 1 external metric.</li></ul>                                  |
| F                        | 1 bit    | Whether a Forwarding Address has been included in the LSA. <ul style="list-style-type: none"><li>● If this field is 1, a Forwarding Address has been included in the LSA.</li><li>● If this field is 0, no Forwarding Address is included in the LSA.</li></ul>   |
| T                        | 1 bit    | Whether an External Route Tag has been included in the LSA. <ul style="list-style-type: none"><li>● If this field is 1, an External Route Tag has been included in the LSA.</li><li>● If this field is 0, no External Route Tag is included in the LSA.</li></ul> |
| Referenced LS Type       | 16 bits  | Referenced LS type. If this value is not 0, an LSA with this LS type is to be associated with this LSA (see Referenced Link State ID below).                                                                                                                      |
| Forwarding Address       | 128 bits | A fully qualified global IPv6 address.                                                                                                                                                                                                                            |
| External Route Tag       | 32 bits  | External route tag, which can be used to communicate additional information between ASBRs.                                                                                                                                                                        |
| Referenced Link State ID | 32 bits  | Referenced link state ID.                                                                                                                                                                                                                                         |

## NSSA-LSA

NSSA-LSAs are originated by ASBRs within an NSSA and describe routes to destinations external to the AS.

[Figure 1-154](#) shows the format of an NSSA-LSA.

**Figure 1-154** Format of an NSSA-LSA

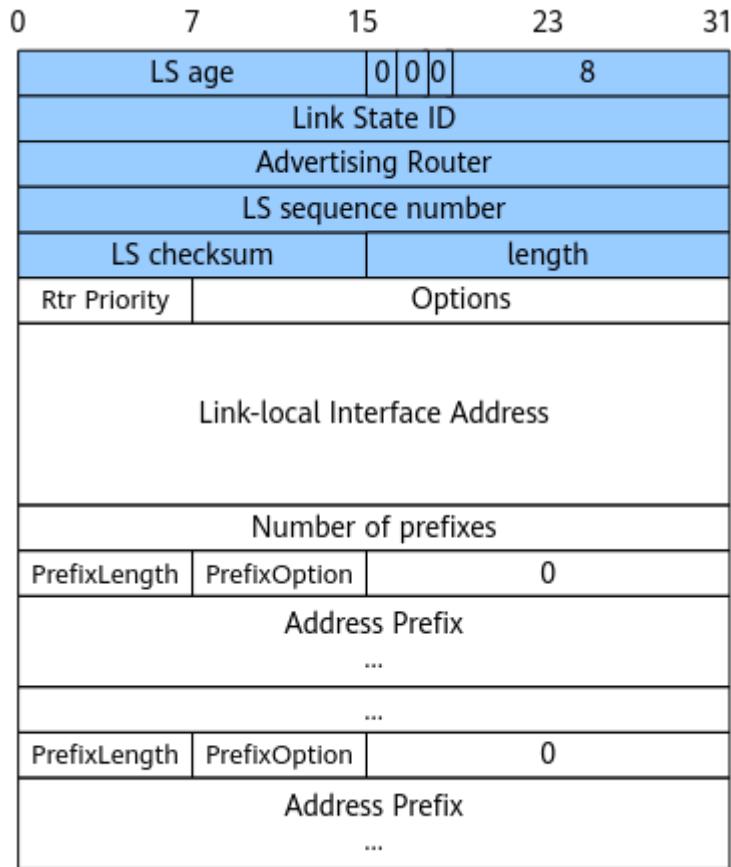


## Link-LSA

Each router generates a link LSA for each link. A link LSA describes the link-local address and IPv6 address prefix associated with the link and the link option set in the network LSA. It is transmitted only on the link.

[Figure 1-155](#) shows the format of a link-LSA.

**Figure 1-155** Format of a Link-LSA



**Table 1-70** Link-LSA fields

| Field                        | Length   | Description                                                                                       |
|------------------------------|----------|---------------------------------------------------------------------------------------------------|
| Rtr Priority                 | 8 bits   | Router priority of the interface.                                                                 |
| Options                      | 24 bits  | Set of options that may be set in the network-LSA generated by the DR on broadcast or NBMA links. |
| Link-local Interface Address | 128 bits | The originating router's link-local interface address on the link.                                |
| Number of prefixes           | 32 bits  | Number of IPv6 address prefixes contained in the LSA.                                             |

## Intra-Area-Prefix-LSA

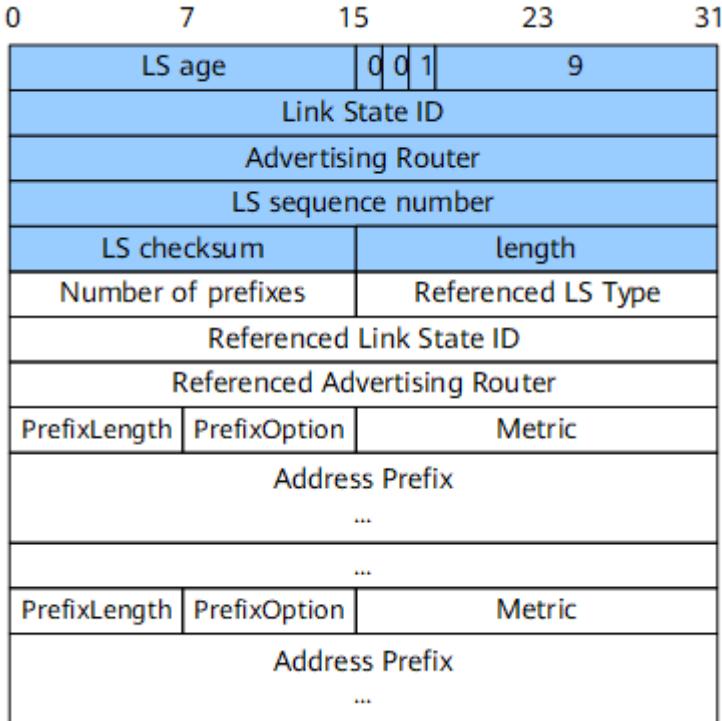
Each router and DR generates one or such LSAs and transmits them in the local area.

- An LSA generated on a router describes the IPv6 address prefix associated with the router LSA.

- Such LSAs generated by a DR describe the IPv6 address prefixes associated with network LSAs.

[Figure 1-156](#) shows the format of an intra-area-prefix-LSA.

**Figure 1-156** Format of an intra-area-prefix-LSA



**Table 1-71** Intra-area-prefix-LSA fields

| Field                    | Length  | Description                                                                                                                                                                                                                                                                                                                       |
|--------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Referenced LS Type       | 16 bits | <p>Router-LSA or network-LSA with which the IPv6 address prefixes should be associated.</p> <ul style="list-style-type: none"> <li>If Referenced LS Type is 0x2001, the IPv6 prefixes are associated with a router-LSA.</li> <li>If Referenced LS Type is 0x2002, the IPv6 prefixes are associated with a network-LSA.</li> </ul> |
| Referenced Link State ID | 32 bits | <p>Referenced link state ID.</p> <ul style="list-style-type: none"> <li>If Referenced LS Type is 0x2001, Referenced Link State ID should be 0.</li> <li>If Referenced LS Type is 0x2002, Referenced Link State ID should be the interface ID of the link's DR.</li> </ul>                                                         |

| Field                         | Length  | Description                                                                                                                                                                                                                                                                                                           |
|-------------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Referenced Advertising Router | 32 bits | <p>ID of the referenced advertising router.</p> <ul style="list-style-type: none"><li>• If Referenced LS Type is 0x2001, Referenced Advertising Router should be the originating router's router ID.</li><li>• If Referenced LS Type is 0x2002, Referenced Advertising Router should be the DR's router ID.</li></ul> |

## BFD for OSPFv3

### Definition

Bidirectional Forwarding Detection (BFD) is a mechanism to detect communication faults between forwarding engines.

To be specific, BFD detects the connectivity of a data protocol along a path between two systems. The path can be a physical link, a logical link, or a tunnel.

In BFD for OSPFv3, a BFD session is associated with OSPFv3. The BFD session quickly detects a link fault and then notifies OSPFv3 of the fault, which speeds up OSPFv3's response to network topology changes.

### Purpose

A link fault or a topology change causes devices to recalculate routes. Therefore, it is important to shorten the convergence time of routing protocols to improve network performance.

As link faults are inevitable, rapidly detecting these faults and notifying routing protocols is an effective way to quickly resolve such issues. If BFD is associated with the routing protocol and a link fault occurs, BFD can speed up the convergence of the routing protocol.

**Table 1-72** BFD for OSPFv3

| With or Without BFD | Link Fault Detection Mechanism | Convergence Speed |
|---------------------|--------------------------------|-------------------|
| Without BFD         | The OSPFv3 Dead timer expires. | Second-level      |
| With BFD            | A BFD session goes Down.       | Millisecond-level |

## Principles

Figure 1-157 BFD for OSPFv3

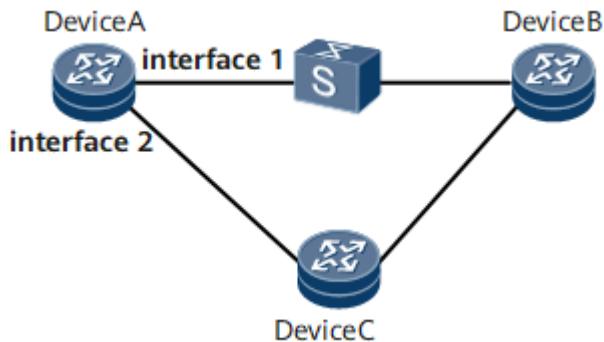


Figure 1-157 shows a typical network topology with BFD for OSPFv3 configured. The principles of BFD for OSPFv3 are described as follows:

1. OSPFv3 neighbor relationships are established between the three devices.
2. After a neighbor relationship becomes Full, a BFD session is established.
3. The outbound interface of the route from DeviceA to DeviceB is interface 1. If the link between DeviceA and DeviceB fails, BFD detects the fault and notifies DeviceA of the fault.
4. DeviceA processes the neighbor Down event and recalculates the route. The new outbound interface of the route is interface 2. Packets from DeviceA pass through DeviceC to reach DeviceB.

## Priority-based Convergence

Priority-based OSPFv3 convergence ensures that some routes converge first if a great number of routes are available. Different routes can be set with different convergence priorities. This allows important routes to converge first and therefore improves network reliability.

With priority-based OSPFv3 convergence, you can set a higher priority for the routes related to key services so that these routes converge first and the impact on key services is minimized.

## OSPFv3 IP FRR

OSPFv3 IP fast reroute (FRR) is dynamic IP FRR, and refers to the process by which OSPFv3 precomputes a backup path based on the network-wide LSDBs, and stores this backup path in the forwarding table. If the primary path fails, traffic can be protected and the fault recovery time is shortened.

OSPFv3 IP FRR complies with standard protocols and provides link and node protection for traffic.

## Background

As networks develop, Voice over Internet Protocol (VoIP) and online video services pose higher requirements for real-time transmission. Nevertheless, if a primary link fails, OSPFv3-enabled devices need to perform multiple operations, including

detecting the fault, updating the LSA, flooding the LSA, calculating routes, and delivering forward information base (FIB) entries before switching traffic to a new link. This process takes a much longer time than the minimum delay to which users are sensitive. As a result, the requirements for real-time transmission cannot be met.

## Fundamentals

OSPFv3 IP FRR refers to a mechanism in which a device uses the loop-free alternate (LFA) algorithm to precompute a backup link, and stores the primary and backup links to the forwarding table. If the primary link fails, OSPFv3 IP FRR switches traffic to the backup link before route convergence is complete on the control plane. This ensures uninterrupted traffic transmission and greatly improves OSPFv3 network reliability. The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports OSPFv3 IP FRR.

A device uses the shortest path first (SPF) algorithm to calculate the shortest path from each neighbor that can provide a backup link to the destination node. The device then uses an inequality defined in the standard protocol to calculate the loop-free backup link with the lowest cost among available shortest paths.

An OSPFv3 IP FRR policy is used to filter backup routes to be added to the IP routing table. Only the backup routes that match the filtering rules of the policy can be added to the IP routing table. Users can configure a desired OSPFv3 IP FRR to filter backup routes.

If a Bidirectional Forwarding Detection (BFD) session is bound to OSPFv3 IP FRR, the BFD session goes down if BFD detects a link fault. If the BFD session goes down, OSPFv3 IP FRR is triggered on the interface to switch traffic from the faulty link to the backup link, which minimizes the loss of traffic.

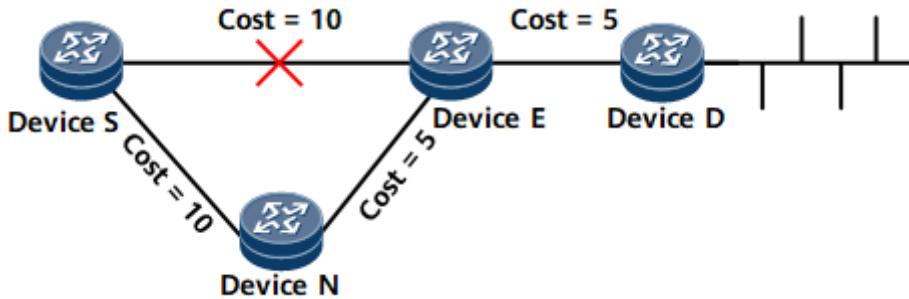
## Usage Scenario

OSPFv3 IP FRR guarantees protection against either a link failure or a node-and-link failure. Distance\_opt (X, Y) indicates the shortest path from node X to node Y.

- **Link protection:** Link protection takes effect when the traffic to be protected flows along a specified link and the link costs meet the inequality:  
$$\text{Distance}_{\text{opt}}(\text{N}, \text{D}) < \text{Distance}_{\text{opt}}(\text{N}, \text{S}) + \text{Distance}_{\text{opt}}(\text{S}, \text{D}).$$
  - S: source node
  - N: node along a backup link
  - D: destination node

On the network shown in [Figure 1-158](#), traffic is forwarded from DeviceS to DeviceD. The primary link is DeviceS -> DeviceE -> DeviceD, and the backup link is DeviceS -> DeviceN -> DeviceE -> DeviceD. The link costs satisfy the link protection inequality. If the primary link fails, DeviceS switches the traffic to the backup link and then traffic can be forwarded to downstream devices, minimizing the traffic interruption duration.

Figure 1-158 Networking for OSPFv3 IP FRR link protection



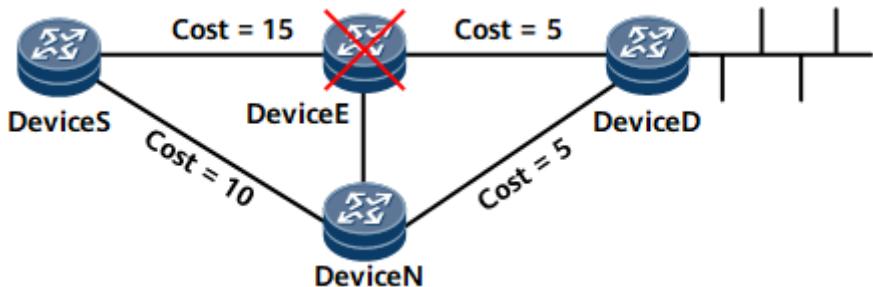
- **Link-and-node protection:** Node-and-link protection takes effect when the traffic to be protected flows along a specified link and node. [Figure 1-159](#) shows the networking for link-and-node protection. The link-and-node protection takes precedence over the link protection.

Link-and-node protection must satisfy the following conditions:

- The link cost must satisfy the inequality:  $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, S) + \text{Distance}_{\text{opt}}(S, D)$ .
- The interface cost must satisfy the inequality:  $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, E) + \text{Distance}_{\text{opt}}(E, D)$ .

S indicates the source node of traffic, E indicates the faulty node, N indicates the node on the backup link, and D indicates the destination node of traffic.

Figure 1-159 Networking for OSPFv3 IP FRR link-and-node protection



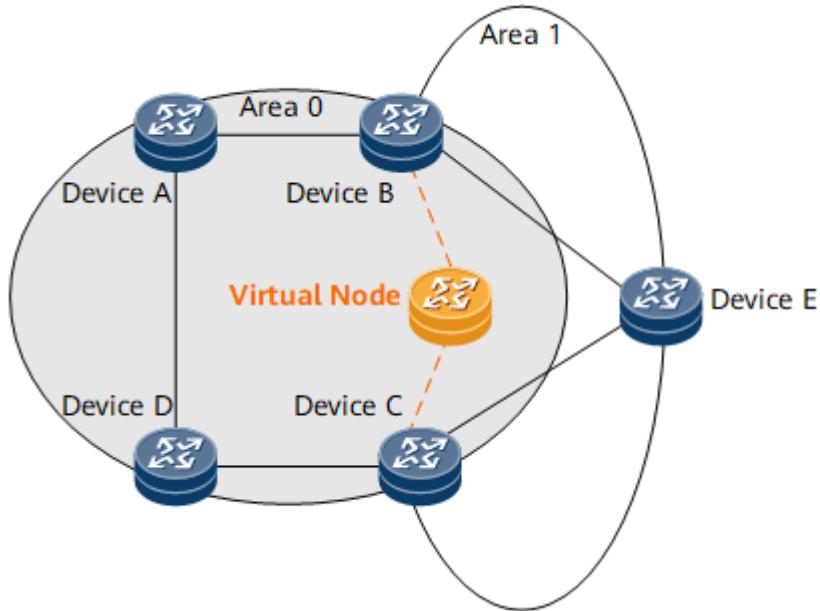
## OSPFv3 FRR in the Scenario Where Multiple Nodes Advertise the Same Route

OSPFv3 IP FRR uses the SPF algorithm to calculate the shortest path from each neighbor (root node) that provides a backup link to the destination node and store the node-based backup next hop, which applies to single-node routing scenarios. As networks are increasingly diversified, two ABRs or ASBRs are deployed to improve network reliability. In this case, OSPFv3 FRR in a scenario where multiple nodes advertise the same route is needed.

 NOTE

In a scenario where multiple nodes advertise the same route, OSPFv3 FRR is implemented by calculating the Type 3 LSAs advertised by ABRs of an area for intra-area, inter-area, and ASE routing. Inter-area routing is used as an example to describe how OSPFv3 FRR works in a scenario where multiple nodes advertise the same route.

**Figure 1-160** OSPFv3 FRR in the scenario where multiple nodes advertise the same route



In **Figure 1-160**, Device B and Device C function as ABRs to forward routes between area 0 and area 1. Device E advertises an intra-area route. Upon receipt of the route, Device B and Device C translate it into a Type 3 LSA and flood the LSA to area 0. After OSPFv3 FRR is enabled on Device A, Device A considers both Device B and Device C as its neighbors. Without a fixed neighbor as the root node, Device A fails to calculate the FRR backup next hop. To address this problem, a virtual node is simulated between Device B and Device C, which converts multi-node routing into single-node routing and uses the LFA algorithm to calculate the backup next hop of the virtual node. The same route advertised by multiple nodes inherits the backup next hop from the virtual node created by the node.

For example, both Device B and Device C advertise the route 2001:DB8:1::1/64, and OSPFv3 FRR is enabled on Device A. After Device A receives the route, it fails to calculate a backup next hop for the route due to a lack of a fixed root node. To address this problem, a virtual node is simulated between Device B and Device C. The virtual node forms a link with each of Device B and Device C. If the virtual node advertises a 2001:DB8:1::1/64 route, it will use the smaller cost of the routes advertised by Device B and Device C as the cost of the route. If the cost of the route advertised by Device B is 5 and that of the route advertised by Device C is 10, the cost of the route advertised by the virtual node is 5. The cost of the link from Device B to the virtual node is 0, and that of the link from Device C to the virtual node is 5. The costs of the links from the virtual node to Device B and Device C are both 65535, the maximum value. Device A is configured to consider Device B and Device C as invalid sources of the 2001:DB8:1::1/64 route and use the

LFA algorithm to calculate the backup next hop for the route, with the virtual node as the root node.

## OSPFv3 GR

### NOTE

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X can be configured as a GR helper rather than a GR restarter.

Graceful Restart (GR) ensures normal traffic forwarding without affecting key services during the restart of routing protocols.

Without GR, the active/standby switchover triggered by any reason leads to transient service interruption, and as a result, route flapping occurs on the whole network. For a large-scale network, the route flapping and service interruption are unacceptable.

GR is a high availability (HA) technology. HA technologies comprise a set of comprehensive techniques, such as fault-tolerant redundancy, link protection, faulty node recovery, and traffic engineering. GR is a fault-tolerant redundancy technology. It has been widely used in active/standby switchover and system upgrade to ensure uninterrupted forwarding of key services.

When the GR function is enabled, the forwarding plane continues data forwarding during a restart, and operations on the control plane, such as re-establishment of neighbor relationships and route calculation, do not affect the forwarding plane, preventing service interruption caused by route flapping and improving network reliability.

## Comparison Between the GR Mode and Non-GR Mode

**Table 1-73** Comparison between the OSPFv3 GR mode and non-GR mode

| Active/Standby Switchover in Non-GR Mode                                                                                                                                                                                                                                                                                                                           | Active/Standby Switchover in GR Mode                                                                                                                                                                                                                                                                                                                                                                       |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"><li>• OSPFv3 neighbor relationships are reestablished.</li><li>• Routes are recalculated.</li><li>• The forwarding table changes.</li><li>• The entire network detects route changes, and route flapping occurs for a short period of time.</li><li>• Traffic is lost during forwarding, and services are interrupted.</li></ul> | <ul style="list-style-type: none"><li>• OSPFv3 neighbor relationships are reestablished.</li><li>• Routes are recalculated.</li><li>• The forwarding table does not change.</li><li>• Except the neighbors of the device on which an active/standby switchover occurs, other devices do not detect route changes.</li><li>• No traffic is lost during forwarding, and services are not affected.</li></ul> |

## OSPFv3 VPN

### Definition

As an extension to OSPFv3, OSPFv3 VPN multi-instance enables Provider Edges (PEs) and Customer Edges (CEs) in VPN scenarios to run OSPFv3 to learn and advertise routes.

### Purpose

As a widely used IGP, in most cases, OSPFv3 runs in VPNs. If OSPFv3 runs between PEs and CEs, and PEs use OSPFv3 to advertise VPN routes to CEs, no other routing protocols need to be configured on CEs for interworking with PEs, which simplifies management and configuration of CEs.

### Running OSPFv3 Between PEs and CEs

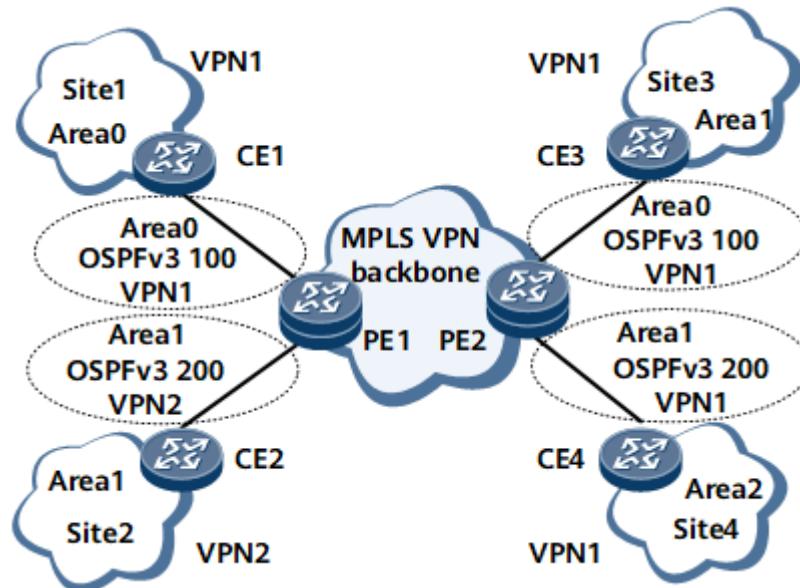
In BGP/MPLS IP VPN, Multi-Protocol BGP (MP-BGP) is used to transmit routing information between PEs, while OSPFv3 is used to learn and advertise routes between PEs and CEs.

Running OSPFv3 between PEs and CEs features the following benefits:

- OSPFv3 is used in a site to learn routes. Running OSPFv3 between PEs and CEs can reduce the number of protocol types supported by CEs.
- Similarly, running OSPFv3 both in a site and between PEs and CEs simplifies the work of network administrators and reduces the number of protocols that network administrators must be familiar with.
- When the network, which originally uses OSPFv3 but not VPN on the backbone network begins to use BGP/MPLS IP VPN, running OSPFv3 between PEs and CEs facilitates the transition.

In [Figure 1-161](#), CE1, CE3, and CE4 belong to VPN 1, and the numbers following OSPFv3 indicate the process IDs of the multiple OSPFv3 instances running on PEs.

**Figure 1-161** Running OSPFv3 between PEs and CEs



CE1 advertises routes to CE3 and CE4 as follows:

1. PE1 imports OSPFv3 routes of CE1 into BGP and forms BGP VPNv6 routes.
2. PE1 uses MP-BGP to advertise the BGP VPNv6 routes to PE2.
3. PE2 imports the BGP VPNv6 routes into OSPFv3 and then advertises these routes to CE3 and CE4.

The processes of advertising routes of CE4 and CE3 to CE1 are the same.

## OSPFv3 Domain ID

If inter-area routes are advertised between local and remote OSPFv3 areas, these areas are considered to be in the same OSPFv3 domain.

- Domain IDs identify and differentiate different domains.
- Each OSPFv3 domain has one or more domain IDs. If more than one domain ID is available, one of the domain IDs is a primary ID, and the others are secondary IDs.
- If an OSPFv3 instance does not have specific domain IDs, its ID is considered as null.

Before advertising the remote routes sent by BGP to CEs, PEs need to determine the type of OSPFv3 routes (Type 3 or Type 5) to be advertised to CEs based on the domain IDs, as shown in **Table 1-74**.

- If local domain IDs are the same as or compatible with the remote domain IDs in BGP routes, Type 3 routes are advertised.
- If local domain IDs are different from or incompatible with the remote domain IDs carried in BGP routes, Type 5 routes are advertised.

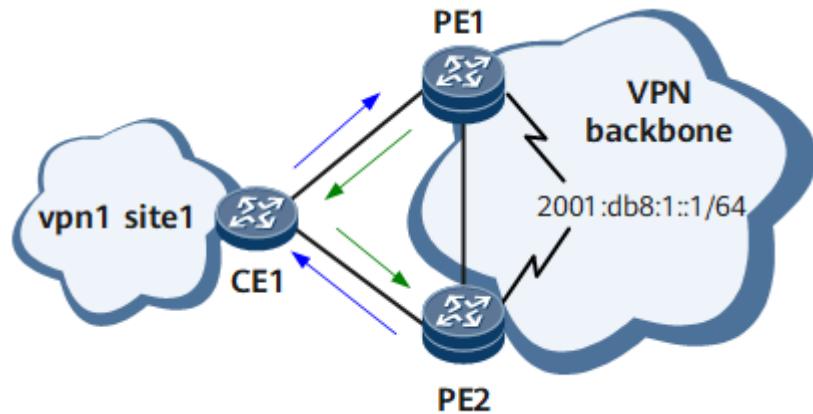
**Table 1-74** Domain ID

| Comparison Between Local and Remote Domain IDs                                                                            | Equal or Not | Route Type                                                                                                                  |
|---------------------------------------------------------------------------------------------------------------------------|--------------|-----------------------------------------------------------------------------------------------------------------------------|
| Both are null.                                                                                                            | Equal        | Inter-area routes                                                                                                           |
| The remote domain ID equals the local primary domain ID or the remote domain ID or one of the local secondary domain IDs. | Equal        | Inter-area routes                                                                                                           |
| The remote domain ID is different from the local primary domain ID or any of the local secondary domain IDs.              | Not equal    | If the local area is a non-NSSA, external routes are generated.<br>If the local area is an NSSA, NSSA routes are generated. |

## Routing Loop Prevention

Routing loops may occur between PEs and CEs when OSPFv3 and BGP learn routes from each other.

Figure 1-162 OSPFv3 VPN routing loops



As shown in [Figure 1-162](#), OSPFv3 on PE1 imports a BGP route destined for 2001:db8:1::1/64, generates a Type 5 or Type 7 LSA, and advertises the LSA to CE1. CE1 learns an OSPFv3 route with the destination address being 2001:db8:1::1/64 and the next hop being PE1, and advertises the route to PE2. In this way, PE2 learns the OSPFv3 route with 2001:db8:1::1/64 as the destination address and CE1 as the next hop.

Similarly, CE1 learns an OSPFv3 route with the destination address being 2001:db8:1::1/64 and next hop being PE2. In this way, PE1 learns the OSPFv3 route with 2001:db8:1::1/64 as the destination address and CE1 as the next hop.

CE1 has two equal-cost routes, with one destined for PE1 and the other destined for PE2. The next hops of the routes from PE1 and PE2 to 2001:db8:1::1/64 are CE1. As a result, a loop occurs.

As OSPFv3 routes have a higher priority than BGP routes, BGP routes to 2001:db8:1::1/64 are replaced by OSPFv3 routes on PE1 and PE2. That is, the OSPFv3 route that is destined for 2001:db8:1::1/64 and has CE1 as the next hop is active in the routing tables of PE1 and PE2.

The BGP route is inactive, and therefore, the LSA generated when this route is imported by OSPFv3 is deleted, which causes the OSPFv3 route to be withdrawn. As a result, no OSPFv3 route exists in the routing table, and the BGP route becomes active again. This cycle causes route flapping.

OSPFv3 VPN provides a few solutions to routing loops, as described in [Table 1-75](#).

**Table 1-75** Routing loop prevention

| Feature Name  | Definition                                                                                                                                                                                                                                                                       | Description                                                                                                                                                                                                                                                                                                                                       |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DN-bit        | OSPFv3 multi-instance processes use a bit flag called the DN bit to prevent routing loops.                                                                                                                                                                                       | When advertising the generated Type 3, Type 5, or Type 7 LSAs to CEs, PEs set the DN-bit of these LSAs to 1 and the DN-bit of other LSAs to 0.<br><br>When calculating routes, the OSPFv3 multi-instance process of the PE ignores LSAs with the DN bit 1. This avoids routing loops that occur when PEs learn the self-originated LSAs from CEs. |
| VPN Route Tag | The VPN route tag is carried in the Type5 or Type7 LSA generated by PEs according to the received BGP private route.<br><br>It is not carried in BGP extended community attributes. The VPN route tag is valid only on the PEs that receive BGP routes and generate OSPFv3 LSAs. | When a PE detects that the VPN route tag in the incoming LSA is the same as that in the local LSA, the PE ignores this LSA to avoid routing loops.                                                                                                                                                                                                |
| Default route | A route with the destination address and mask of all 0s is a default route.                                                                                                                                                                                                      | PEs do not calculate default routes.<br><br>Default routes forward traffic from CEs or sites where CEs reside to the VPN backbone network.                                                                                                                                                                                                        |

## Multi-VPN-Instance CE

OSPFv3 multi-instance generally runs on PEs. Devices that run OSPFv3 multi-instance within user LANs are called Multi-VPN-Instance CEs (MCEs).

Compared with OSPFv3 multi-instance running on PEs, MCEs have the following characteristics:

- MCEs do not need to support OSPFv3-BGP association.
- MCEs establish one OSPFv3 instance for each service. Different virtual CEs transmit different services, which ensures LAN security at a low cost.
- MCEs implement different OSPFv3 instances on a CE. The key to implementing MCEs is to disable loop detection and calculate routes directly.

That is, MCEs also use the received LSAs with the DN bit set to 1 for route calculation.

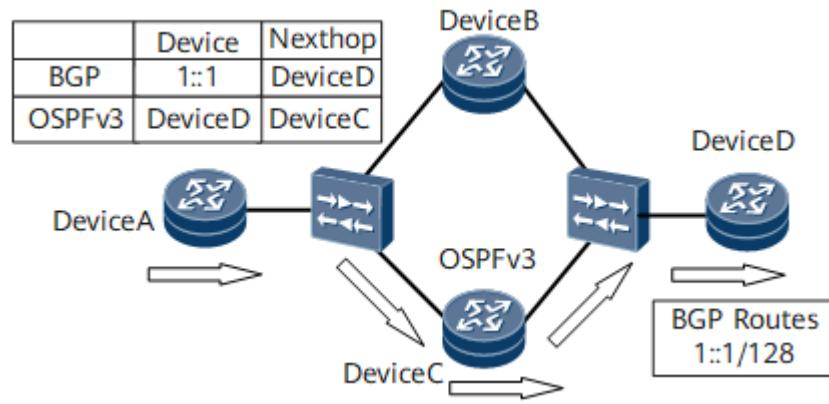
## OSPFv3-BGP Association

New device deployment or a device restart may lead to network traffic loss during BGP convergence. This is due to IGP convergence being faster than BGP convergence. OSPFv3-BGP association can address this problem.

After a device on a BGP network recovers from a fault, BGP convergence is performed again and packet loss may occur during the convergence.

As shown in [Figure 1-163](#), traffic from DeviceA to DeviceD traverses a BGP network through DeviceC.

**Figure 1-163** Traffic traversing a BGP network

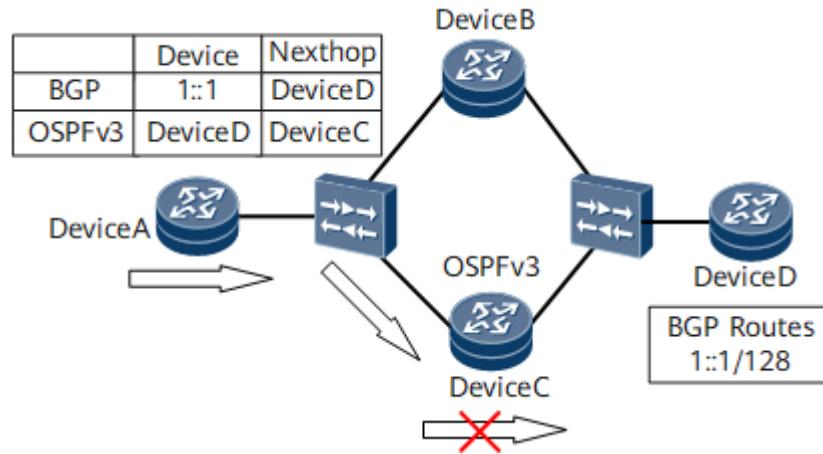


If DeviceC fails, traffic passes through DeviceB due to route reselection. Packets are lost when DeviceC recovers.

Because OSPFv3 route convergence is faster than BGP route convergence, OSPFv3 convergence completes first when DeviceC recovers. On DeviceA, the next hop of the route to DeviceD is DeviceC. However, BGP convergence on DeviceC is not complete. As a result, DeviceC does not know how to reach DeviceD.

In this manner, when traffic from DeviceA to DeviceD is sent to DeviceC, DeviceC discards the traffic because it does not have the route to DeviceD, as shown in [Figure 1-164](#).

**Figure 1-164** Packet loss upon device restart without OSPFv3-BGP association



## Process of OSPFv3-BGP Association

A device enabled with OSPFv3-BGP association sets the maximum link cost (65535) in its LSA to ensure that the device is not used as a traversal device. BGP routes, however, can still reach the device.

As shown in [Figure 1-163](#), OSPFv3-BGP association is enabled on DeviceC. Before BGP convergence is complete, DeviceA does not forward traffic to DeviceC. Instead, DeviceA continues to use the backup link (passing through DeviceB) until BGP routes on DeviceC are converged.

## OSPFv3 Authentication

### OSPFv3 IPsec Authentication

The rapid development of networks poses higher requirements for network security. Routing protocol packets that are transmitted on networks may be intercepted, changed, or forged, and packet attacks may cause network interruption. Therefore, packets need to be protected.

The standard protocol does not define the authentication mechanism for OSPFv3. Therefore, OSPFv3 packets do not carry any authentication information. However, the standard protocol defines the IPsec mechanism to authenticate OSPFv3 packets.

The IPsec protocol family, which consists of a series of protocols defined by the Internet Engineering Task Force (IETF), provides high-quality, interoperable, and cryptology-based security for IP packets. By encrypting data and authenticating the data source at the IP layer, communicating parties can ensure confidentiality, data integrity, data authentication, and anti-replay for the data transmitted across the network.

 NOTE

- Confidentiality: The data is encrypted and transmitted in cipher text.
- Data integrity: Received data is authenticated to check whether they have been modified.
- Data authentication: The data source is authenticated to ensure that the data is sent from a real sender.
- Anti-replay: The attacks from malicious users who repeatedly send obtained data packets are prevented. Specifically, the receiver rejects old or repeated data packets.

IPsec adopts two security protocols: Authentication Header (AH) and Encapsulating Security Payload (ESP).

- AH: A protocol that provides data origin authentication, data integrity check, and anti-replay protection. AH does not encrypt packets to be protected.  
AH data is carried in the following fields:
  - Version number
  - Packet header length
  - Packet length
  - Authentication
  - Protocol
  - Source and destination addresses
  - Options
- ESP: A protocol that provides IP packet encryption mechanism besides the functions provided by AH. The encryption and authentication mechanisms can be used together or independently.

## OSPFv3 Authentication Trailer

Prior to the OSPFv3 Authentication Trailer, OSPFv3 can use only IPsec for authentication. However, on some special networks, a mobile ad hoc network (MANET) for example, IPsec is difficult to deploy and maintain. To address this problem, Authentication Trailer for OSPFv3 can be used to implement authentication.

OSPFv3 authentication encrypts OSPFv3 packets by adding the authentication field to packets to ensure network security. A local device checks the authentication field in OSPFv3 packets received from a remote device, and discards the packets if they do not contain the same authentication password as the locally configured one, thereby achieving self-protection.

In terms of authentication scope, OSPFv3 authentication is classified as follows:

- Area authentication  
This authentication mode is configured in the OSPFv3 area view and applies to packets received by all interfaces in the OSPFv3 area.
- Process authentication  
This authentication mode is configured in the OSPFv3 view and applies to all packets in the OSPFv3 process.
- Interface authentication

This authentication mode is configured in the interface view and applies to all packets received by the interface.

OSPFv3 uses HMAC-SHA256 to authenticate packets. In HMAC-SHA256 authentication, a configured password is hashed using the HMAC-SHA256 algorithm, and the ciphertext password is added to packets for authentication. This authentication mode improves password security.

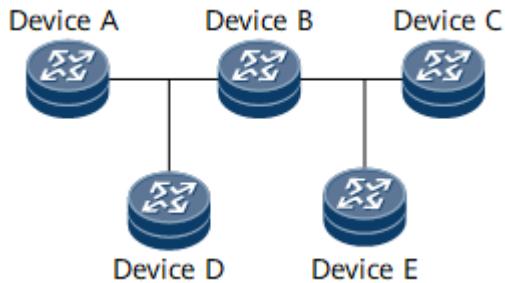
OSPFv3 carries authentication types in packet headers and authentication information in packet trailers.

The authentication types are as follows:

- 1: Simple authentication
- 2: Ciphertext authentication

## Networking Application of OSPFv3 Authentication Trailer

Figure 1-165 OSPFv3 authentication trailer on a broadcast network



Configuration requirements:

- The interface authentication configurations must be the same on all devices on the same network so that OSPFv3 neighbor relationships can be established.
- Area authentication configurations must be the same on all devices in the same area.

## OSPFv3 Neighbor Relationship Flapping Suppression

OSPFv3 neighbor relationship flapping suppression works by delaying OSPFv3 neighbor relationship reestablishment or setting the link cost to the maximum value.

## Background

If an interface carrying OSPFv3 services frequently alternates between up and down, frequent neighbor relationship flapping will occur. During the flapping, OSPFv3 frequently sends Hello packets to reestablish the neighbor relationship, synchronizes LSDBs, and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, OSPFv3 services, and other OSPFv3-dependent services, such as LDP and BGP. OSPFv3 neighbor relationship flapping suppression can address this problem by delaying

OSPFv3 neighbor relationship reestablishment or preventing service traffic from passing through flapping links.

## Related Concepts

**flapping\_event**: reported when the status of a neighbor relationship on an interface last changes from Full to a non-Full state. The flapping\_event triggers flapping detection.

**flapping\_count**: number of times flapping has occurred.

**detect-interval**: detection interval. The interval is used to determine whether to trigger a valid flapping\_event.

**threshold**: flapping suppression threshold. When the flapping\_count reaches or exceeds **threshold**, flapping suppression takes effect.

**resume-interval**: interval for exiting from OSPFv3 neighbor relationship flapping suppression. If the interval between two successive valid flapping\_events is longer than **resume-interval**, the flapping\_count is reset.

## Implementation

### Flapping detection

Each OSPFv3 interface on which OSPFv3 neighbor relationship flapping suppression is enabled starts a **flapping\_count**. If the interval between two successive neighbor status changes from Full to a non-Full state is shorter than **detect-interval**, a valid flapping\_event is recorded, and the **flapping\_count** increases by 1. When the **flapping\_count** reaches or exceeds **threshold**, flapping suppression takes effect. If the interval between two successive neighbor status changes from Full to a non-Full state is longer than **resume-interval**, the **flapping\_count** is reset.

The **detect-interval**, **threshold**, and **resume-interval** are configurable.

#### NOTE

The value of **resume-interval** must be greater than that of **detect-interval**.

### Flapping suppression

Flapping suppression works in either Hold-down or Hold-max-cost mode.

- Hold-down mode: In the case of frequent flooding and topology changes during neighbor relationship establishment, interfaces prevent neighbor relationships from being reestablished during the suppression period, which minimizes LSDB synchronization attempts and packet exchanges.
- Hold-max-cost mode: If the traffic forwarding path changes frequently, interfaces use 65535 as the cost of the flapping link during Hold-max-cost suppression, which prevents traffic from passing through the flapping link.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

By default, the Hold-max-cost mode takes effect. The mode and suppression duration can be changed manually.

If an attack causes frequent neighbor relationship flapping, Hold-down mode can minimize the impact of the attack.

 **NOTE**

When an interface enters the flapping suppression state, all neighbor relationships on the interface enter the state accordingly.

### Exiting from flapping suppression

Interfaces exit from flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding OSPFv3 process is reset.
- An OSPFv3 neighbor relationship is reset.
- A command is run to exit from flapping suppression.

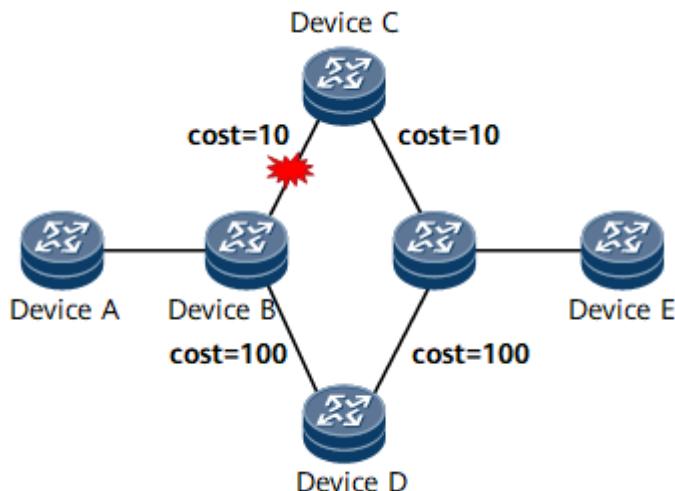
## Typical Scenarios

### Basic scenario

In [Figure 1-166](#), the traffic forwarding path is Device A -> Device B -> Device C -> Device E before a link failure occurs. After the link between Device B and Device C fails, the forwarding path switches to Device A -> Device B -> Device D -> Device E. If the neighbor relationship between Device B and Device C frequently flaps at the early stage of the path switchover, the forwarding path will be switched frequently, causing traffic loss and affecting network stability. If the neighbor flapping between Device B and Device C meets flapping suppression conditions, flapping suppression is triggered.

- If flapping suppression works in Hold-down mode, the neighbor relationship between Device B and Device C is prevented from being reestablished during the suppression period, in which traffic is forwarded along the path Device A -> Device B -> Device D -> Device E.
- If flapping suppression works in Hold-max-cost mode, 65535 is used as the cost of the link between Device B and Device C during the suppression period, and traffic is forwarded along the path Device A -> Device B -> Device D -> Device E.

**Figure 1-166** Flapping suppression in a basic scenario



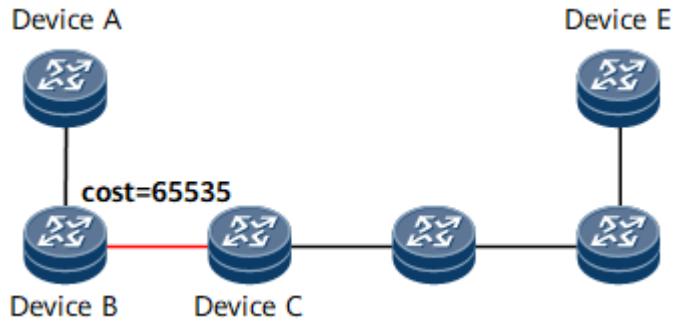
### Single-forwarding path scenario

When only one forwarding path exists on the network, the flapping of the neighbor relationship between any two devices on the path will interrupt traffic forwarding. In [Figure 1-167](#), the traffic forwarding path is Device A -> Device B -> Device C -> Device E. If the neighbor relationship between Device B and Device C flaps, and the flapping meets suppression conditions, flapping suppression takes effect. However, if the neighbor relationship between Device B and Device C is prevented from being reestablished, the whole network will be divided. Therefore, Hold-max-cost mode (rather than Hold-down mode) is recommended. If flapping suppression works in Hold-max-cost mode, 65535 is used as the cost of the link between Device B and Device C during the suppression period. After the network stabilizes and the suppression timer expires, the link is restored.

#### NOTE

By default, the Hold-max-cost mode takes effect.

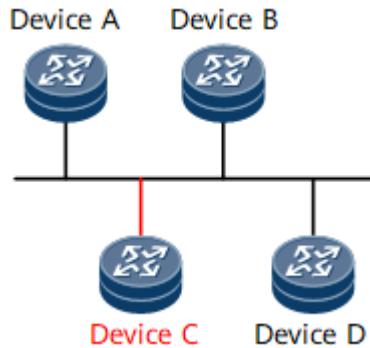
**Figure 1-167** Flapping suppression in a single-forwarding path scenario



### Broadcast scenario

In [Figure 1-168](#), four devices are deployed on the same broadcast network using switches, and the devices are broadcast network neighbors. If Device C flaps due to a link failure, and Device A and Device B were deployed at different time (Device A was deployed earlier for example) or the flapping suppression parameters on Device A and Device B are different, Device A first detects the flapping and suppresses Device C. Consequently, the Hello packets sent by Device A do not carry Device C's router ID. However, Device B has not detected the flapping yet and still considers Device C a valid node. As a result, the DR candidates identified by Device A are Device B and Device D, whereas the DR candidates identified by Device B are Device A, Device C, and Device D. Different DR candidates result in a different DR election result, which may lead to route calculation errors. In scenarios where an interface has multiple neighbors, such as on a broadcast, P2MP, or NBMA network, all neighbors on the interface are suppressed if the last flapping event that the neighbor status on the interface goes down is detected (flapping detection cannot be performed by neighbor). Specifically, if Device C flaps, Device A, Device B, and Device D on the broadcast network are all suppressed. After the network stabilizes and the suppression timer expires, Device A, Device B, and Device D are restored to normal status.

Figure 1-168 Flapping suppression on a broadcast network



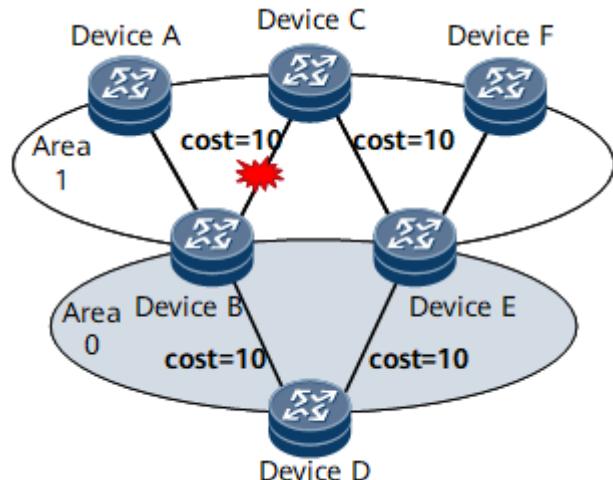
### Multi-area scenario

In [Figure 1-169](#), Device A, Device B, Device C, Device E, and Device F are connected in area 1, and Device B, Device D, and Device E are connected in backbone area 0. Traffic from Device A to Device F is preferentially forwarded along an intra-area route, and the forwarding path is Device A -> Device B -> Device C -> Device E -> Device F. If the neighbor relationship between Device B and Device C flaps and the flapping meets suppression conditions, flapping suppression takes effect in the default mode (Hold-max-cost). However, the forwarding path remains unchanged (Device A -> Device B -> Device C -> Device E -> Device F) after the neighbor flapping occurs because intra-area routes take precedence over inter-area routes during route selection regardless of costs according to OSPF route selection rules. The Hold-max-cost mode cannot suppress traffic path switching in this case. To prevent traffic loss in multi-area scenarios, configure Hold-down mode to prevent the neighbor relationship between Device B and Device C from being reestablished during the suppression period. During this period, traffic is forwarded along the path Device A -> Device B -> Device D -> Device E -> Device F.

#### NOTE

By default, the Hold-max-cost mode takes effect. The mode can be changed to Hold-down manually.

Figure 1-169 Flapping suppression in a multi-area scenario



## OSPFv3 Flush Source Tracing

### Context

If network-wide OSPFv3 LSA flush causes network instability, source tracing must be implemented as soon as possible to locate and isolate the fault source. However, OSPFv3 itself does not support source tracing. A conventional solution is isolation node by node until the faulty node is located. The solution is complex and time-consuming, which also affects user network services. To solve the preceding problem, OSPFv3 introduces a proprietary protocol, namely, the source tracing protocol. This protocol supports the flooding of flush source information. When the preceding problem occurs, you can quickly query the flush source information on any device on the network to quickly locate the fault source.

### Related Concepts

#### Source tracing

A mechanism that helps locate the device that flushes OSPFv3 LSAs. It has the following features:

- Uses a new UDP port. Source tracing packets are carried by UDP packets, and the UDP packets carry the OSPFv3 LSAs flushed by the current device and are flooded hop by hop based on the OSPFv3 topology.
- Depends on OSPFv3 neighbors for flooding and forwards packets along UDP channels, which are independent of the channels used to transmit OSPFv3 packets. Therefore, this protocol facilitates incremental deployment. In addition, source tracing does not affect the devices with the related UDP port disabled.
- Supports query of the node that flushed LSAs on any device that supports this feature after source tracing packets are flooded on the network, which speeds up fault locating and faulty node isolation by maintenance personnel.

#### Flush

Network-wide OSPFv3 LSAs are deleted.

#### PS-Hello packets

Packets used to negotiate the OSPFv3 flush source tracing capability between OSPFv3 neighbors.

#### PS-LSA

When a device flushes an OSPFv3 LSA, it generates a PS-LSA carrying information about the device and brief information about the OSPFv3 LSA.

#### PS-LSU packets

OSPFv3 flush source tracing packets that carry PS-LSAs.

#### PS-LSU ACK packets

Acknowledgment packets used to enhance the reliability of OSPFv3 flush source tracing packets.

#### OSPFv3 flush source tracing port

ID of the UDP port that receives and sends OSPFv3 flush source tracing packets.  
The ID is configurable.

## Fundamentals

OSPFv3 flush source tracing is implemented as follows:

1. Source tracing capability negotiation

After an OSPFv3 neighbor relationship is established between two devices, they need to negotiate the source tracing capability through PS-Hello packets.

2. PS-LSA generation and flooding

When a device flushes an OSPFv3 LSA, it generates a PS-LSA carrying information about the device and brief information about the OSPFv3 LSA, adds the PS-LSA to a PS-LSU packet, and floods the PS-LSU packet to source tracing-capable neighbors, which helps other devices locate the fault source and perform isolation.

 NOTE

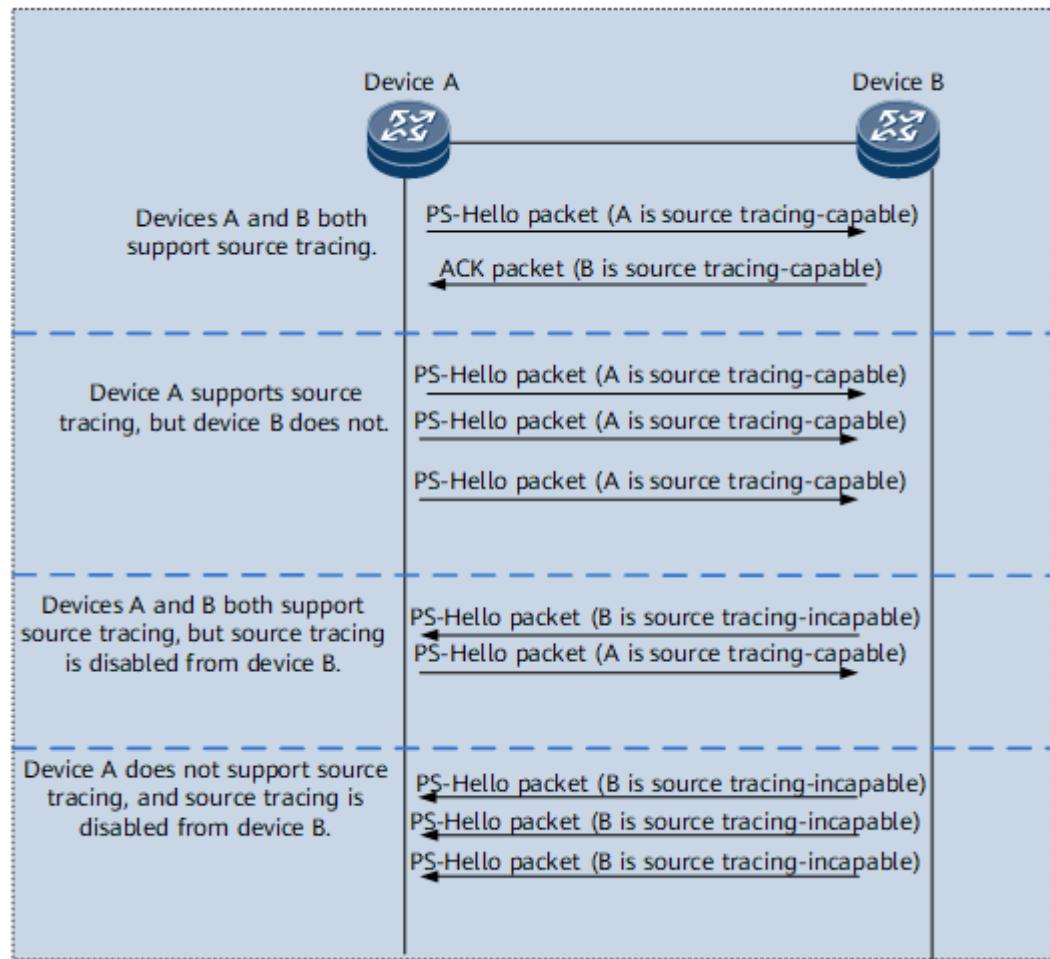
Only router-LSAs, network-LSAs, and inter-area-router-LSAs can be flushed. Therefore, a device generates a PS-LSA only when it flushes a router-LSA, network-LSA, or inter-area-router-LSA.

## Source Tracing Capability Negotiation

The source tracing protocol uses UDP to carry the proprietary protocol packets and listens to the UDP port to receive and send source tracing packets. If a source tracing-capable Huawei device sends source tracing packets to a source tracing-incapable Huawei device or non-Huawei device, the source tracing-capable Huawei device may be incorrectly identified as an attacker. Therefore, the source tracing capability needs to be negotiated between the devices. In addition, the source tracing-capable device needs to send source tracing information on behalf of the source tracing-incapable device, which also requires negotiation.

Source tracing depends on OSPFv3. Therefore, source tracing negotiation depends on OSPFv3 neighbor relationships. Specifically, after an OSPFv3 neighbor relationship is established, the local device initiates source tracing capability negotiation. [Figure 1-170](#) shows the negotiation process.

**Figure 1-170** Source tracing capability negotiation



**Table 1-76** Source tracing capability negotiation

| Whether Source Tracing Is Supported          | Source Tracing Capability Negotiation Process                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Devices A and B both support source tracing. | <ol style="list-style-type: none"> <li>1. DeviceA sends a PS-Hello packet to notify its source tracing capability.</li> <li>2. Upon reception of the PS-Hello packet, DeviceB sets the source tracing field for DeviceA and replies with an ACK packet to notify its source tracing capability to DeviceA.</li> <li>3. Upon reception of the ACK packet, DeviceA sets the source tracing field for DeviceB, and does not retransmit the PS-Hello packet.</li> </ol> |

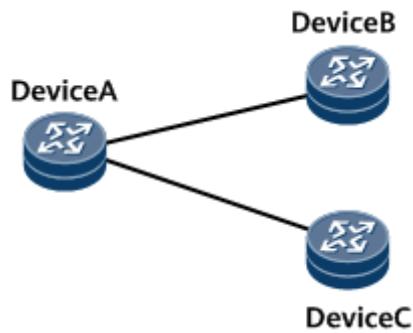
| Whether Source Tracing Is Supported                                                       | Source Tracing Capability Negotiation Process                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| DeviceA supports source tracing, but DeviceB does not.                                    | <ol style="list-style-type: none"> <li>1. DeviceA sends a PS-Hello packet to notify its source tracing capability.</li> <li>2. DeviceA fails to receive an ACK packet from DeviceB within 10s and retransmits the PS-Hello packet. A maximum of two retransmissions are allowed. After DeviceA fails to receive an ACK packet from DeviceB after two retransmissions, DeviceA considers that DeviceB does not support source tracing.</li> </ol>                          |
| Devices A and B both support source tracing, but source tracing is disabled from DeviceB. | <ol style="list-style-type: none"> <li>1. After source tracing is disabled from DeviceB, DeviceB sends a PS-Hello packet to notify its source tracing incapability.</li> <li>2. Upon reception of the PS-Hello packet from DeviceB, DeviceA replies with an ACK packet that carries the source tracing capability.</li> <li>3. Upon reception of the ACK packet from DeviceA, DeviceB considers the capability negotiation complete and disables the UDP port.</li> </ol> |
| DeviceA does not support source tracing, and source tracing is disabled from DeviceB.     | <ol style="list-style-type: none"> <li>1. After source tracing is disabled from DeviceB, DeviceB sends a PS-Hello packet to notify its source tracing incapability.</li> <li>2. DeviceB fails to receive an ACK packet within 10s and retransmits the PS-Hello packet. A maximum of two retransmissions are allowed. After two retransmissions, DeviceB considers the capability negotiation complete and disables the UDP port.</li> </ol>                               |

## PS-LSA Generation and Flooding

PS-LSA: carries information about the node that flushed OSPFv3 LSAs.

- If a device flushes an LSA, it generates and floods a PS-LSA to source tracing-capable neighbors.
- If a device receives a flush LSA from a source tracing-incapable neighbor, the device generates and floods a PS-LSA to source tracing-capable neighbors. If a device receives the same flush LSA (with the same LSID and sequence number) from more than one source tracing-incapable neighbors, the device generates only one PS-LSA.
- If a device flushes a router-LSA, network-LSA, or inter-area-router-LSA, it generates a PS-LSA, adds the PS-LSA to a PS-LSU packet, and floods the PS-LSU packet to all source tracing-capable neighbors.

Figure 1-171 PS-LSA generation rules



### PS-LSA generation rules

- When DeviceA flushes a router-LSA, network-LSA, or inter-area-router-LSA, it generates a PS-LSA in which the **Flush Router** field is its router ID and the **Neighbor Router** field is 0, and adds the PS-LSA to the queue where packets are to be sent to all source tracing-capable neighbors.
- After DeviceA receives the flush LSA from source tracing-incapable DeviceB, DeviceA generates a PS-LSA in which the **Flush Router** field is its router ID and the **Neighbor Router** field is the router ID of DeviceB, and adds the PS-LSA to the queue where packets are to be sent to all source tracing-capable neighbors.
- After DeviceA receives the flush LSA from DeviceB, followed by the same flush LSA sent by DeviceC, DeviceA generates a PS-LSA in which the **Flush Router** field is its router ID and the **Neighbor Router** field is the router ID of DeviceB, and adds the PS-LSA to the queue where packets are to be sent to all source tracing-capable neighbors. No PS-LSA is generated in response to the flush LSA received from DeviceC.

### PS-LSU packet sending rules

- During neighbor relationship establishment, a device initializes the sequence number of the PS-LSU packet of the neighbor. When the device replies with a PS-LSU packet, it adds the sequence number of the PS-LSU packet of the neighbor. During PS-LSU packet retransmission, the sequence number remains unchanged. After the device receives a PS-LSU ACK packet with the same sequence number, it increases the sequence number of the neighbor's PS-LSU packet by 1.
- The neighbor manages the PS-LSA sending queue. When a PS-LSA is added to the queue which was empty, the neighbor starts a timer. After the timer expires, the neighbor adds the PS-LSA to a PS-LSU packet, sends the packet to its neighbor, and starts another timer to wait for a PS-LSU ACK packet.
- After the PS-LSU ACK timer expires, the PS-LSU packet is retransmitted.
- When the device receives a PS-LSU ACK packet with a sequence number same as that in the neighbor record, the device clears PS-LSAs from the neighbor queue, and sends another PS-LSU packet after the timer expires.
  - If the sequence number of a received PS-LSU ACK packet is less than that in the neighbor record, the device ignores the packet.
  - If the sequence number of a received PS-LSU ACK packet is greater than that in the neighbor record, the device discards the packet.

 NOTE

PS-LSU packet sending is independent among neighbors.

**PS-LSU packet receiving rules**

- When a device receives a PS-LSU packet from a neighbor, the neighbor records the sequence number of the packet and replies with a PS-LSU ACK packet.
- When the device receives a PS-LSU packet with the sequence number the same as that in the neighbor record, the device discards the PS-LSU packet.
- After the device parses a PS-LSU packet, it adds the PS-LSA in the packet to the LSDB. The device also checks whether the PS-LSA is newer than the corresponding PS-LSA in the LSDB.
  - If the received PS-LSA is newer, the device floods it to other neighbors.
  - If the received PS-LSA is the same as the corresponding local one, the device does not process the received PS-LSA.
  - If the received PS-LSA is older, the device floods the corresponding PS-LSA in the LSDB to the neighbor.
- If the device receives a PS-LSU packet from a neighbor and the neighbor does not support source tracing, the device modifies the neighbor status as source tracing capable.

## Source Tracing Security

The source tracing protocol uses UDP packets, and a private port needs to be enabled to send and receive protocol packets. Therefore, the security of the port must be considered.

The source tracing protocol inevitably increases packet receiving and sending workload and intensifies bandwidth pressure. To minimize its impact on original protocols and reduce the traffic of source tracing packets, the number of source tracing packets must be controlled.

Table 2 describes source tracing security measures.

**Table 1-77** Security measures for source tracing

| Security Measures for Source Tracing | Fundamentals                                                                                                                                                |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Authentication                       | Source tracing is embedded in OSPFv3, inherits existing OSPFv3 configuration parameters, and uses OSPFv3 authentication parameters to authenticate packets. |

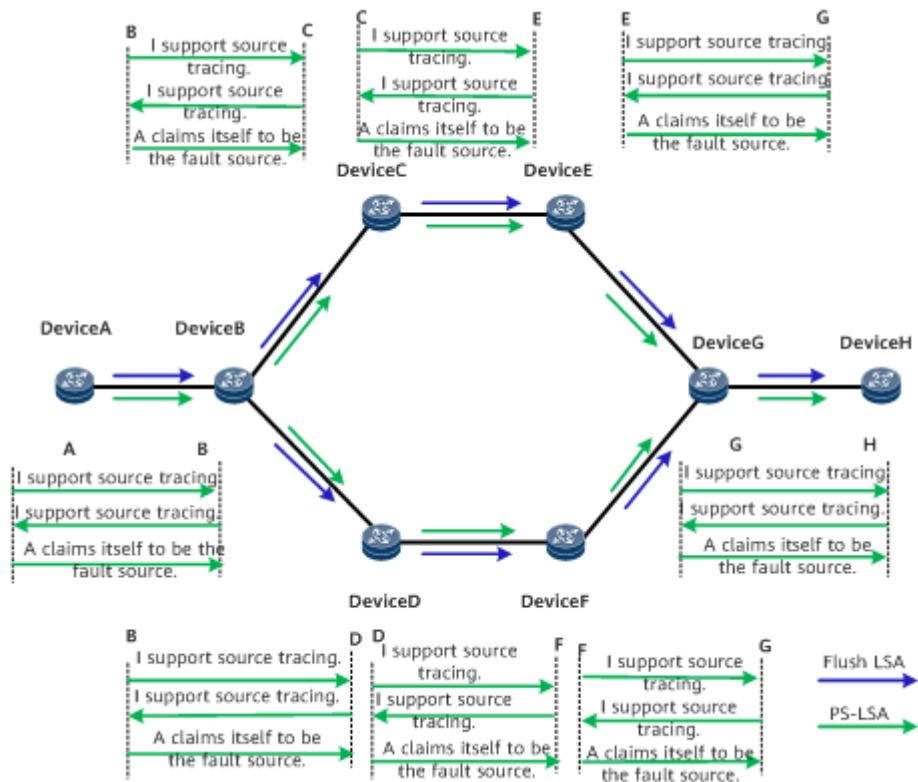
| Security Measures for Source Tracing | Fundamentals                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GTSM                                 | <p>GTSM is a security protocol that checks whether the time to live (TTL) value in each sent and received IP packet is within a pre-defined range.</p> <p>Source tracing packets can only be flooded as far as one hop. Therefore, GTSM can be used to check such packets by default.</p> <ul style="list-style-type: none"><li>• When a device sends a packet, it sets the TTL of the packet to 255.</li><li>• If the TTL is not 254 when the packet is received, the packet will be discarded.</li></ul> |
| CPU-CAR                              | <p>Interface boards can check the packets to be sent to the CPU for processing and prevent the main control board from being overloaded by a large number of packets that are sent to the CPU. The source tracing protocol needs to apply for an independent CAR channel and has small CAR values configured.</p>                                                                                                                                                                                          |

## Typical Scenarios

### Scenario where all nodes support source tracing

Assume that all nodes on the network support source tracing and DeviceA is the fault source. In this scenario, the fault source can be accurately located. [Figure 1-172](#) shows the networking.

**Figure 1-172 Scenario where all nodes support source tracing**

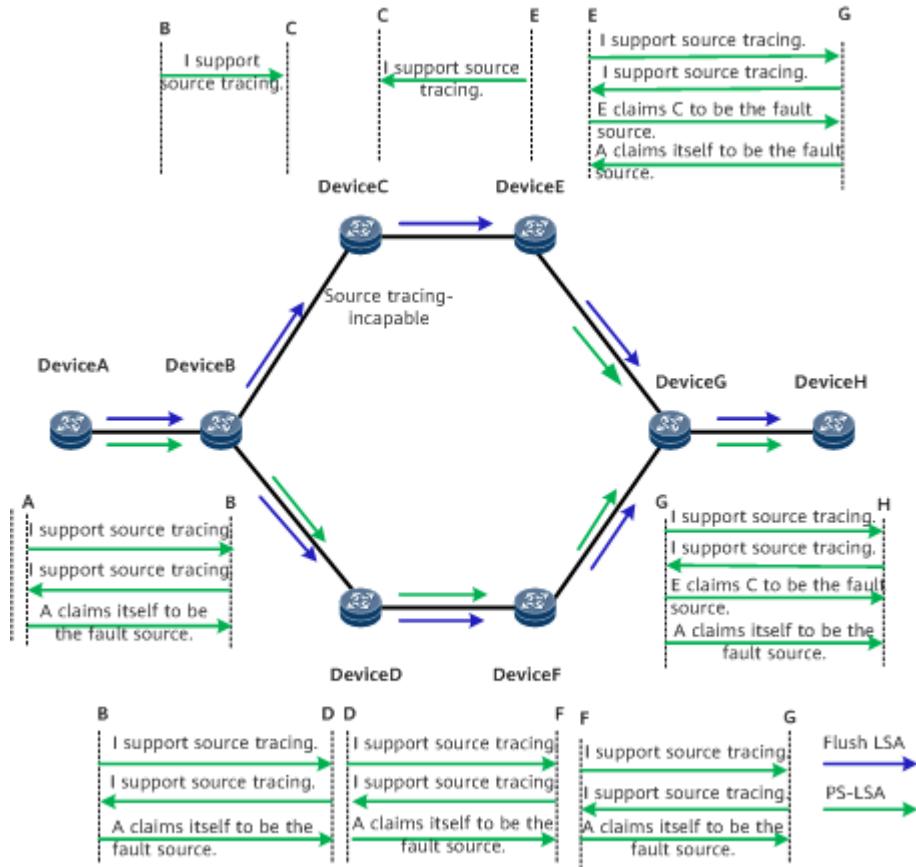


When DeviceA flushes an OSPFv3 LSA, it generates a PS-LSA that carries DeviceA information and brief information about the OSPFv3 flush LSA. Device A then floods the PS-LSA. After the fault occurs, maintenance personnel can log in to any node on the network to locate DeviceA, which keeps sending flush LSAs, and isolate DeviceA from the network.

#### Scenario where source tracing-incapable nodes are not isolated from source tracing-capable nodes

All nodes on the network except DeviceC support source tracing, and DeviceA is the fault source. In this case, the PS-LSA can be flooded on the entire network, and the fault source can be accurately located. [Figure 1-173](#) shows the networking.

**Figure 1-173 Scenario where source tracing-incapable nodes are not isolated from source tracing-capable nodes**



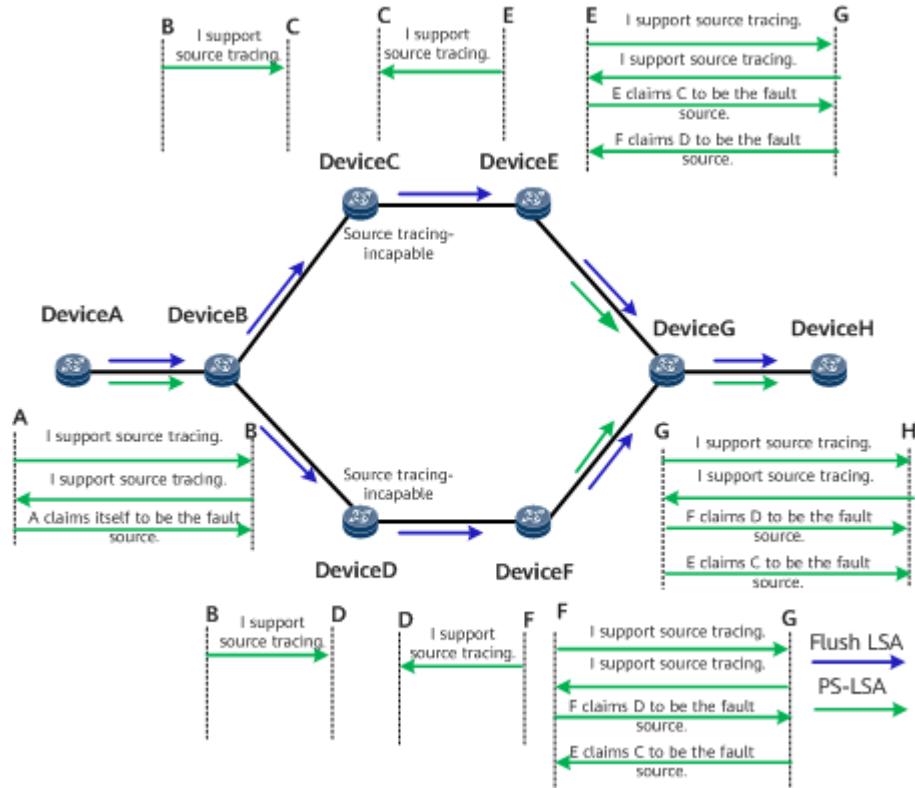
When DeviceA flushes an OSPFv3 LSA, it generates a PS-LSA that carries DeviceA information and brief information about the OSPFv3 flush LSA. Device A then floods the PS-LSA. When DeviceB and DeviceE negotiate the source tracing capability with DeviceC, they find that DeviceC does not support source tracing. Therefore, after DeviceB receives the PS-LSA from DeviceA, DeviceB sends the PS-LSA to DeviceD, but not to DeviceC. Device A sends a flush LSA to Device B. When Device C sends the flush LSA to Device E, Device E finds that Device C does not support source tracing, helps Device C generate a PS-LSA that carries information about the advertisement source (Device E) and flush source (Device C), and flushed OSPFv3 LSA, and floods the PS-LSA on the entire network.

After the fault occurs, maintenance personnel can log in to any device on the network except DeviceC to locate the fault source. Two possible fault sources can be located in this case. One fault source is advertised by Device A, which claims that it has initiated a large number of flush packets. The other fault source is advertised by Device E and it claims that Device C may initiate a large number of flush packets. Both Device A and Device C flush the same LSA. In this case, DeviceA takes precedence over DeviceC. Therefore, Device A is the fault source and measures are taken to isolate Device A. After DeviceA is isolated, the network recovers.

#### Scenario where source tracing-incapable nodes are isolated from source tracing-capable nodes

All nodes on the network except DeviceC and DeviceD support source tracing, and DeviceA is the fault source. In this case, the PS-LSA cannot be flooded on the entire network, and the fault source can only be found within a certain range. [Figure 1-174](#) shows the networking.

**Figure 1-174** Scenario where source tracing-incapable nodes are isolated from source tracing-capable nodes



When DeviceA flushes an OSPFv3 LSA, it generates a PS-LSA that carries DeviceA information and brief information about the OSPFv3 flush LSA. Device A then floods the PS-LSA. However, the PS-LSA can reach only DeviceB because DeviceC and DeviceD do not support source tracing.

During source tracing capability negotiation, DeviceE finds that DeviceC does not support source tracing, and DeviceF finds that DeviceD does not support source tracing. After DeviceE receives the flush LSA from DeviceC, DeviceE generates and floods a PS-LSA on behalf of DeviceC. Similarly, after DeviceF receives the flush LSA from DeviceD, DeviceF generates and floods a PS-LSA on behalf of DeviceD.

After the fault occurs:

- If maintenance personnel log in to DeviceA or DeviceB, the personnel can locate the fault source (DeviceA) directly. After DeviceA is isolated, the network recovers.
- If the maintenance personnel log in to DeviceE, DeviceF, DeviceG, or DeviceH, the personnel will find that DeviceE claims DeviceC to be the fault source of the OSPFv3 flush LSA and DeviceF claims DeviceD to be the fault source of the same OSPFv3 flush LSA.

- If the maintenance personnel log in to DeviceC and DeviceD, the personnel will find that the flush LSA was initiated by DeviceB, not generated by DeviceC or DeviceD.
- If the maintenance personnel log in to DeviceB, the personnel will find that DeviceA is the fault source, and isolate DeviceA. After DeviceA is isolated, the network recovers.

## Routing Loop Detection for Routes Imported to OSPFv3

Routes of an OSPFv3 process can be imported to another OSPFv3 process or the process of another protocol (such as IS-IS or BGP) for redistribution. However, if a device that performs such a route import is incorrectly configured, routing loops may occur. OSPFv3 can use the routing loop detection function to detect routing loops.

## Related Concepts

### Redistribute ID

IS-IS uses a system ID as a redistribution identifier, OSPF and OSPFv3 use a router ID + process ID as a redistribution identifier, and BGP uses a VrfID + random number as a redistribution identifier. For ease of understanding, the redistribution identifiers of different protocols are all called Redistribute IDs. When routes are distributed, the information carried in the routes contains Redistribute IDs.

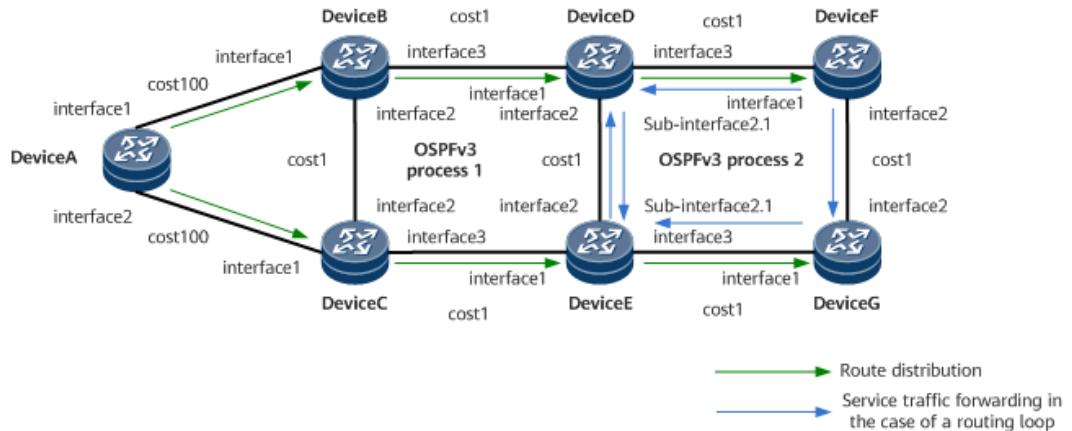
### Redistribute List

A Redistribute list may consist of multiple Redistribute IDs. Each Redistribute list of BGP contains a maximum of four Redistribute IDs, and each Redistribute list of any other routing protocol contains a maximum of two Redistribute IDs. When the number of Redistribute IDs exceeds the corresponding limit, the old ones are discarded according to the sequence in which Redistribute IDs are added.

## Background (OSPFv3 Inter-Process Mutual Route Import)

On the network shown in [Figure 1-175](#), DeviceA, DeviceB, and DeviceC run OSPFv3 process 1; DeviceF and DeviceG run OSPFv3 process 2; DeviceD and DeviceE run both of the processes. Route import between OSPFv3 process 1 and OSPFv3 process 2 is configured on DeviceD and DeviceE. The routes distributed by OSPFv3 process 1 on DeviceE are re-distributed back to OSPFv3 process 1 on DeviceD through OSPFv3 process 2. As the costs of the routes newly distributed by DeviceD are smaller, they are preferentially selected by OSPFv3 process 1, resulting in routing loops.

**Figure 1-175** Typical network diagram of OSPFv3 inter-process mutual route import

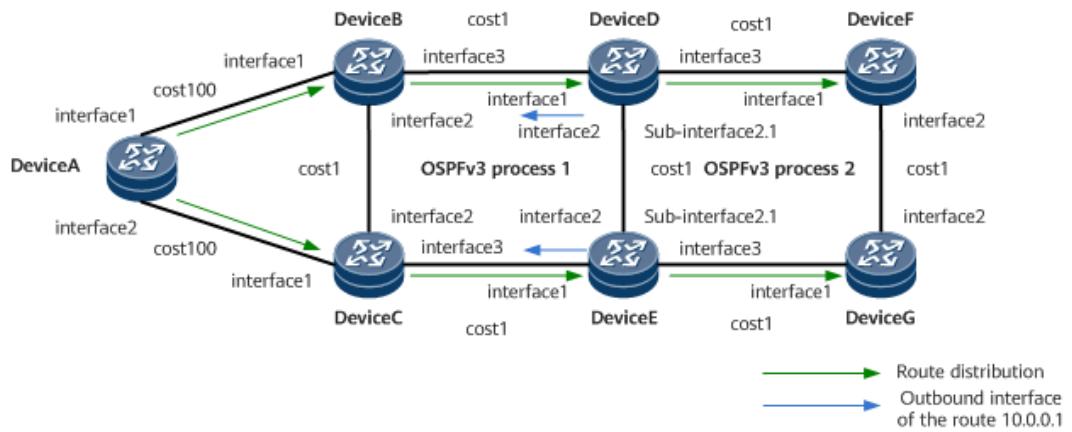


Take the route distributed by DeviceA as an example. A stable routing loop is formed through the following process:

### Phase 1

On the network shown in [Figure 1-176](#), OSPFv3 process 1 on DeviceA imports the static route 10.0.0.1 and floods a Type 5 AS-External-LSA in OSPFv3 process 1. After receiving the LSA, OSPFv3 process 1 on DeviceD and OSPFv3 process 1 on DeviceE each calculate a route to 10.0.0.1, with the outbound interfaces being interface1 on DeviceD and interface1 on DeviceE, respectively, and the cost being 102. At this point, the routes to 10.0.0.1 in OSPFv3 process 1 in the routing tables of DeviceD and DeviceE are active.

**Figure 1-176** Phase 1

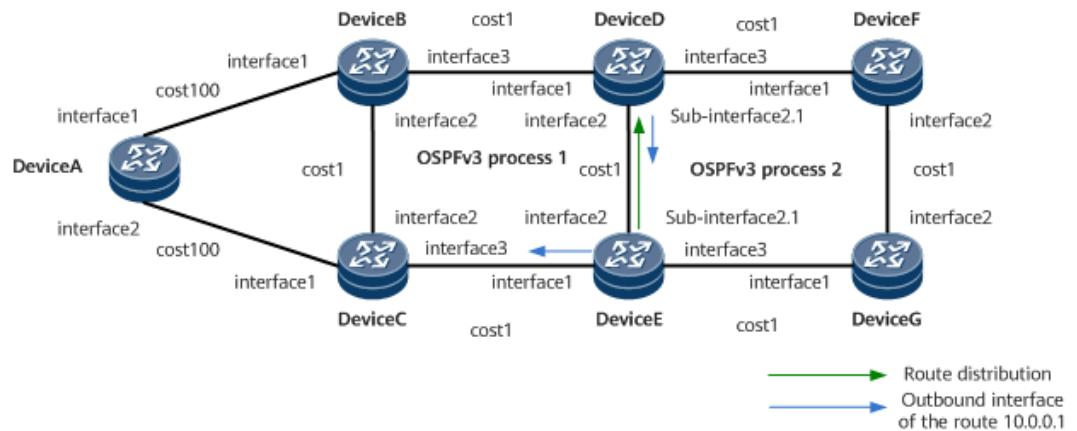


### Phase 2

In [Figure 1-177](#), DeviceD and DeviceE are configured to import routes from OSPFv3 process 1 to OSPFv3 process 2. No route-policy is configured for the import, or the configured route-policy is improper. For example, OSPFv3 process 2 on DeviceE imports routes from OSPFv3 process 1 and then floods a Type 5 AS-External-LSA in OSPFv3 process 2. After receiving the LSA, OSPFv3 process 2 on DeviceD calculates a route to 10.0.0.1, with the cost being 2, which is smaller than that (102) of the route calculated by OSPFv3 process 1. As a result, the active

route to 10.0.0.1 in the routing table of DeviceD is switched from the one calculated by OSPFv3 process 1 to the one calculated by OSPFv3 process 2, and the outbound interface of the route is sub-interface2.1.

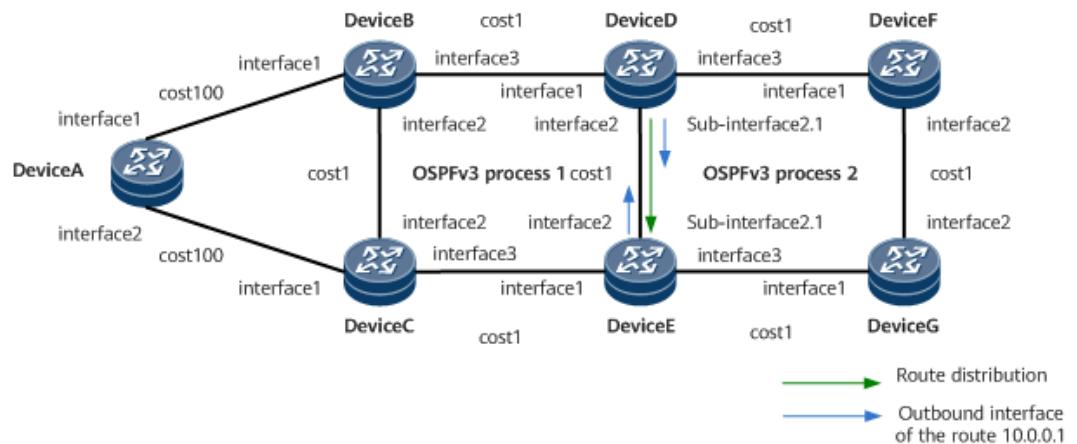
**Figure 1-177 Phase 2**



### Phase 3

In [Figure 1-178](#), DeviceD imports the route from OSPFv3 process 2 to OSPFv3 process 1 and floods a Type 5 AS-External LSA in OSPFv3 process 1. After receiving the LSA, OSPFv3 process 1 on DeviceE recalculates the route to 10.0.0.1. The cost of the route becomes 2, which is smaller than that of the previously calculated route. Therefore, the route to 10.0.0.1 in OSPFv3 process 1 on DeviceE is changed to the route distributed by DeviceD, and the outbound interface is interface 2.

**Figure 1-178 Phase 3**

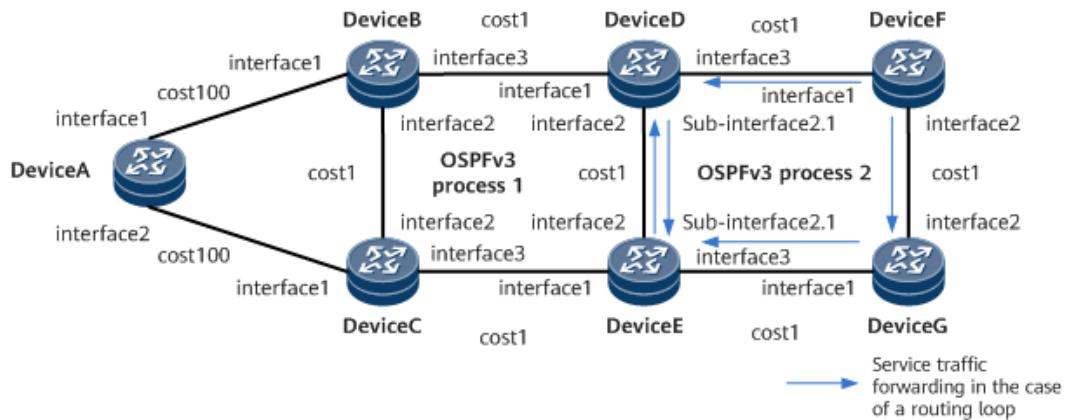


### Phase 4

After the route to 10.0.0.1 on DeviceE is updated, OSPFv3 process 2 still imports the route from OSPFv3 process 1 as the route remains active, and continues to distribute/update a Type 5 AS-External-LSA.

As a result, a stable routing loop is formed. Assuming that traffic is injected from DeviceF, [Figure 1-179](#) shows the traffic flow when the routing loop occurs.

**Figure 1-179** Traffic flow when a routing loop occurs

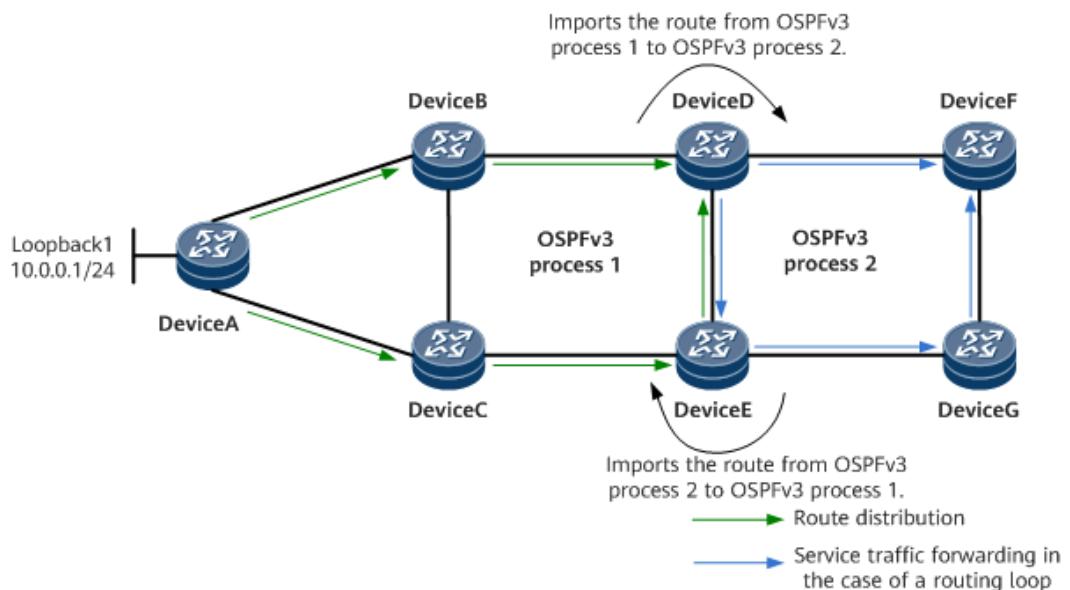


## Implementation (OSPFv3 Inter-Process Mutual Route Import)

Routing loop detection for the routes imported between OSPFv3 processes can resolve the routing loops in the preceding scenario.

When distributing a Type 5 AS-External LSA for an imported route, OSPFv3 also uses an E-AS-External-LSA to distribute to other devices the Redistribute ID of the device that redistributes the imported route. If the route is redistributed by different protocols through multiple devices, all the Redistribute IDs of these protocols on the devices are distributed through an E-AS-External-LSA. When receiving the E-AS-External-LSA, a route calculation device saves the Redistribute ID and route information of the route redistribution device. When another process of a route calculation device imports the route, the device checks whether a routing loop occurs according to the route redistribution information. If a routing loop occurs, the device attaches a large route cost to the AS-External-LSA for the imported route so that other devices preferentially select other paths after learning the route. This prevents routing loops.

**Figure 1-180** Typical networking of route import to OSPFv3



**Figure 1-180** is used to describe how a routing loop is detected and resolved.

1. DeviceA distributes its locally originated route 10.0.0.1/24 to DeviceB.
2. DeviceD learns the route distributed by DeviceB through OSPFv3 process 1 and imports the route from OSPFv3 process 1 to OSPFv3 process 2. DeviceE learns the route distributed by DeviceD through OSPFv3 process 2 and saves the Redistribute List distributed by DeviceD through OSPFv3 process 2 to the routing table when calculating routes.
3. DeviceE imports the route from OSPFv3 process 2 to OSPFv3 process 1 and redistributes the route through OSPFv3 process 1. The corresponding E-AS-External-LSA contains the Redistribute ID of OSPFv3 process 1 on DeviceE and the Redistribute ID of OSPFv3 process 2 on DeviceD. The Redistribute ID of OSPFv3 process 1 on DeviceB has been discarded from the LSA.
4. OSPFv3 process 1 on DeviceD learns the Redistribute list corresponding to the route distributed by DeviceE and saves the Redistribute list in the routing table. When importing the route from OSPFv3 process 1 to OSPFv3 process 2, DeviceD finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. OSPFv3 process 2 on DeviceD distributes a large cost when redistributing the route so that other devices preferentially select other paths after learning the route. This prevents routing loops.

 **NOTE**

In the preceding typical networking:

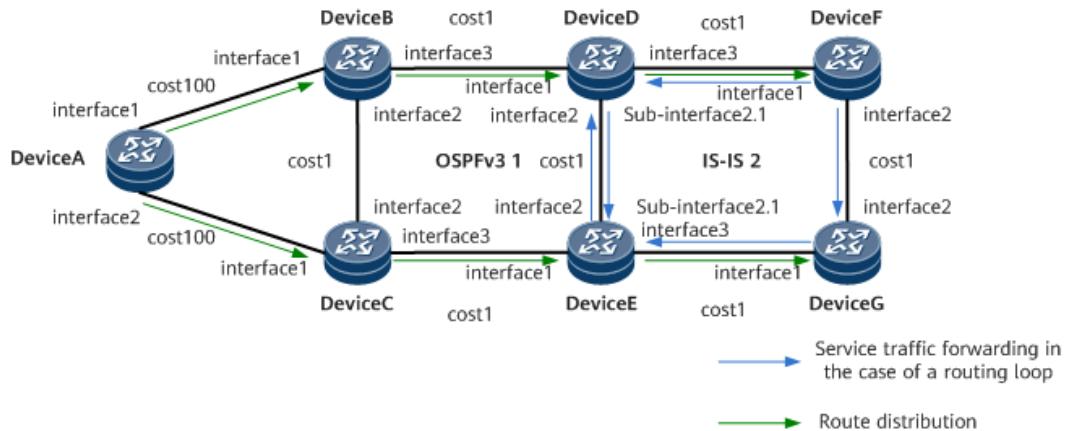
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

## Background (Mutual Route Import Between OSPFv3 and IS-IS)

On the network shown in **Figure 1-181**, DeviceA, DeviceB, and DeviceC run OSPFv3 process 1, DeviceF and DeviceG run IS-IS process 2, and DeviceD and DeviceE run both processes. Route import between OSPFv3 process 1 and IS-IS process 2 is configured on DeviceD and DeviceE. The routes distributed by OSPFv3 process 1 on DeviceE are re-distributed back to OSPFv3 process 1 on DeviceD through IS-IS process 2. As the costs of the routes newly distributed by DeviceD are smaller, they are preferentially selected by OSPFv3 process 1, resulting in routing loops.

**Figure 1-181** Traffic flow when a routing loop occurs during route import between OSPFv3 and IS-IS



## Implementation (Mutual Route Import Between OSPFv3 and IS-IS)

The following uses the networking shown in [Figure 1-181](#) as an example to describe how a routing loop is detected and resolved.

1. DeviceD learns the route distributed by DeviceB through OSPFv3 process 1 and imports the route from OSPFv3 process 1 to IS-IS process 2. When IS-IS process 2 on DeviceD distributes route information, it uses the extended prefix sub-TLV to distribute the Redistribute ID of IS-IS process 2 through an LSP. IS-IS process 2 on DeviceE learns the route distributed by DeviceD and saves the Redistribute ID distributed by IS-IS process 2 on DeviceD to the routing table during route calculation.
2. DeviceE imports the route from IS-IS process 2 to OSPFv3 process 1 and uses an E-AS-External-LSA to distribute the Redistribute ID of OSPFv3 process 1 on DeviceE when redistributing the imported route. Similarly, after OSPFv3 process 1 on DeviceD learns the route from DeviceE, DeviceD saves the Redistribute ID distributed by OSPFv3 process 1 on DeviceE to the routing table during route calculation.
3. When importing the route from OSPFv3 process 1 to IS-IS process 2, DeviceD finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. IS-IS process 2 on DeviceD distributes a large cost when distributing the imported route. Because IS-IS has a higher preference than OSPFv3 ASE, this does not affect the route selection result or resolve the routing loop.
4. DeviceE imports the route from IS-IS process 2 to OSPFv3 process 1, finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. OSPFv3 process 1 on DeviceE distributes a large cost when distributing the imported route so that other devices preferentially select other paths after learning the route. This prevents routing loops.

 NOTE

In the preceding typical networking:

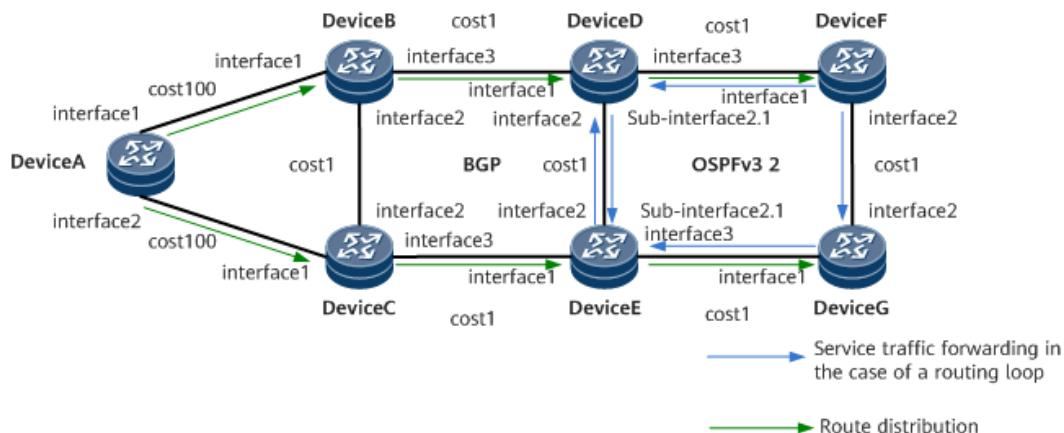
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

## Background (Mutual Route Import Between OSPFv3 and BGP)

On the network shown in [Figure 1-182](#), DeviceA, DeviceB, and DeviceC run a BGP process, DeviceF and DeviceG run OSPFv3 process 2, and DeviceD and DeviceE run both processes. Route import between BGP and OSPFv3 process 2 is configured on DeviceD and DeviceE. The routes distributed by BGP on DeviceE are redistributed back to BGP through OSPFv3 process 2 on DeviceD. Because no route-policy is configured for the import or the configured route-policy is improper, the route newly distributed by DeviceD may be selected as the optimal route by BGP, causing a routing loop.

**Figure 1-182** Traffic flow when a routing loop occurs during route import between OSPFv3 and BGP



## Implementation (Mutual Route Import Between OSPFv3 and BGP)

The following uses the networking shown in [Figure 1-182](#) as an example to describe how a routing loop is detected and resolved.

1. DeviceD learns the route distributed by DeviceB through BGP and imports the BGP route to OSPFv3 process 2. When DeviceD distributes the imported route through OSPFv3 process 2, it uses an extended prefix E-AS-External-LSA to distribute the Redistribute ID of OSPFv3 process 2 on DeviceD. DeviceE learns the route distributed by DeviceD through OSPFv3 process 2 and saves the Redistribute List distributed by DeviceD through OSPFv3 process 2 to the routing table when calculating routes.

2. DeviceE imports the route from OSPFv3 process 2 to BGP and distributes the Redistribute ID of the BGP process on DeviceE through an E-AS-External-LSA when redistributing the imported route. After BGP on DeviceD learns the route distributed by DeviceE, DeviceD saves the Redistribute ID distributed by BGP on DeviceE to the routing table during route calculation.
3. When importing the route from BGP to OSPFv3 process 2, DeviceD finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. OSPFv3 process 2 on DeviceD distributes a large link cost when distributing the imported route. Because OSPFv3 has a higher preference than BGP, this does not affect the route selection result or resolve the routing loop.
4. When importing the route from OSPFv3 process 2 to BGP, DeviceE finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. In addition, when BGP on DeviceE distributes the imported route, it reduces the preference of the route. In this way, other devices preferentially select other paths after learning this route, preventing routing loops.

 NOTE

In the preceding typical networking:

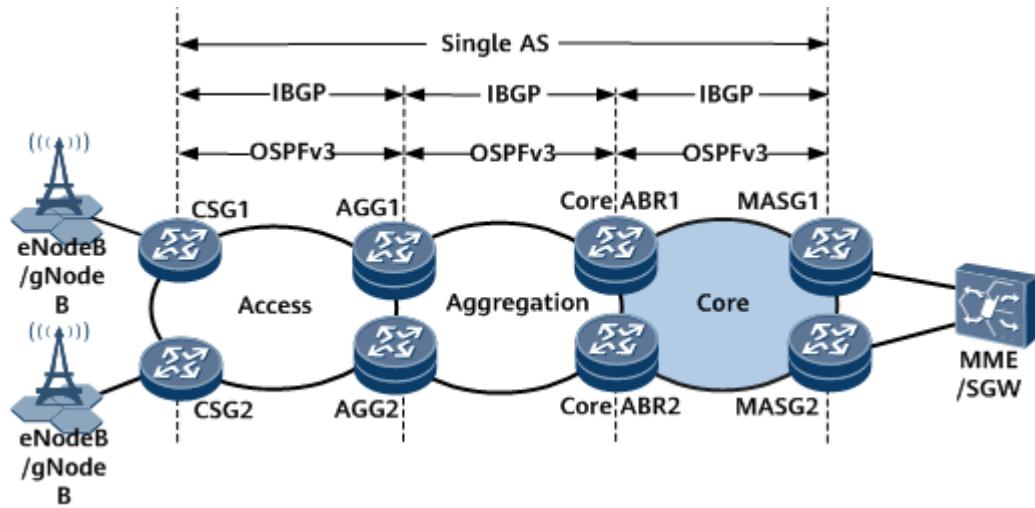
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

## Usage Scenario

**Figure 1-183** shows a typical seamless MPLS network. If the OSPFv3 process deployed at the access layer differs from that deployed at the aggregation layer, OSPFv3 inter-process mutual route import is usually configured on AGGs so that routes can be leaked between the access and aggregation layers. As a result, a routing loop may occur between AGG1 and AGG2. If routing loop detection for OSPFv3 is configured on AGG1 and AGG2, routing loops can be quickly detected and other routes can be preferentially selected, preventing routing loops.

**Figure 1-183** Routing protocol deployment on the intra-AS seamless MPLS network



### 1.1.5.2 OSPFv3 Configuration

This chapter describes the key concepts of Open Shortest Path First version 3 (OSPFv3) and provides an overview of its configuration and maintenance. OSPFv3 is applicable to large-scale networks with hundreds of devices.

#### 1.1.5.2.1 OSPFv3 Overview

OSPFv3 uses the same basic implementation mechanism as OSPFv2 but is incompatible with OSPFv2.

#### Definition

Open Shortest Path First (OSPF) is a link-state Interior Gateway Protocol (IGP) developed by the Internet Engineering Task Force (IETF).

OSPF version 2 (OSPFv2) is intended for IPv4, and OSPF version 3 (OSPFv3) is intended for IPv6.

- OSPFv3 is short for OSPF version 3.
- OSPFv3 runs over IPv6.
- OSPFv3 is an enhanced version of OSPFv2 but is an independent routing protocol.

#### Purpose

OSPFv3 is an extension of OSPF for support of IPv6.

### 1.1.5.2.2 Configuration Precautions for OSPFv3

#### Feature Requirements

**Table 1-78** Feature requirements

| Feature Requirements                                                                                                                                                                                                                                                                                                                                          | Series           | Models                                                                                             |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|----------------------------------------------------------------------------------------------------|
| OSPFv3 checks whether a neighbor in the full state exists in the backbone area before advertising a default route to the stub area.                                                                                                                                                                                                                           | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/<br>NetEngine 8000 X16/NetEngine 8000E X8/<br>NetEngine 8100 X |
| The network types of the interfaces on both ends of an OSPFv3 neighbor relationship must be the same so that the interfaces can learn routes after the neighbor relationship is established. Otherwise, the neighbor relationship cannot be established and routes cannot be calculated.<br><br>You are advised to use the same network type at both ends.    | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/<br>NetEngine 8000 X16/NetEngine 8000E X8/<br>NetEngine 8100 X |
| If the OSPFv3 neighbor timeout period is too short, the neighbor relationship may go Down due to timeout, affecting services.<br><br>You are advised to use the default hello timer and dead timer, or change the dead timer to a value greater than or equal to 40s.                                                                                         | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/<br>NetEngine 8000 X16/NetEngine 8000E X8/<br>NetEngine 8100 X |
| After the device is restarted, if the BFD session status of the local device or neighbor is AdminDown, the OSPFv3 status is not affected. When the BFD session is renegotiated, if BFD reports the detection status Down but the previous detection status is Up, OSPFv3 sets the neighbor status to Down. In other cases, the OSPFv3 status is not affected. | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/<br>NetEngine 8000 X16/NetEngine 8000E X8/<br>NetEngine 8100 X |

| Feature Requirements                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Series           | Models                                                                                     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| <ol style="list-style-type: none"><li>1. A maximum of 100,000 route processing loops are supported. If the number of imported route advertisement extension LSAs exceeds the upper limit, the LSAs are not advertised.</li><li>2. Routes cannot be imported to NSSAs.</li><li>3. Aggregated routes cannot advertise redistribute information.</li><li>4. After a loop is removed, the loop cannot be automatically removed. You need to manually clear the alarm using a command.</li><li>5. If the redistribute IDs of different devices conflict, a loop alarm may be falsely reported.</li><li>6. Loop detection is not supported in the scenario where more than two re-advertisement nodes are configured.</li><li>7. Loops may fail to be detected in load balancing scenarios where inner-loop routes and non-loop routes are advertised by multiple nodes.</li><li>8. If the original route source is withdrawn and a loop is triggered, self-healing is not supported.</li><li>9. Route import between public and private networks is not supported.</li><li>10. After detecting a route loop, the re-advertisement point increases the cost of the route to prevent the loop. As a result, the traffic direction may be different from the expected path.</li><li>11. When the maximum cost is advertised on a loop, the apply cost command does not take effect.</li><li>12. After this function is enabled, LSAs need to be added to advertise extended TLVs for each prefix, which greatly increases memory usage.</li></ol> | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

### 1.1.5.2.3 Configuring Basic OSPFv3 Functions

To construct an OSPFv3 network, configure basic OSPFv3 functions first.

#### Usage Scenario

OSPFv3 functions take effect only after an OSPFv3 process, the router ID, the interface, and the area ID are specified.

OSPFv3 functions can be configured in the interface view, regardless of whether OSPFv3 is enabled. If OSPFv3 is disabled from the interface, OSPFv3 functions configured on this interface are still valid.

## Pre-configuration Tasks

Before configuring basic OSPFv3 functions, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Enable IPv6.

## (Optional) Configuring OSPFv3 Short-Timeout Neighbor Enhancement

OSPFv3 short-timeout neighbor enhancement improves neighbor stability in scenarios where there are many short-timeout neighbors.

## Context

If the configured interval at which an interface sends Hello packets is less than 10s, many OSPFv3 short-timeout neighbors may exist. (For details about setting the interval, see [Setting the Interval at Which Hello Packets Are Sent](#).) In this case, if the device has poor performance, and the CPU usage is high or a switchover is performed, the neighbors may be unstable due to scheduling issues. To improve the stability of short-timeout neighbors in scenarios where there are more than 150 OSPF and OSPFv3 short-timeout neighbors, you can configure the short-timeout neighbor enhancement function, which increases the OSPFv3 short-timeout neighbor specification.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run ospf short-timeout neighbor enhancement

OSPFv3 short-timeout neighbor enhancement is configured.

#### NOTE

- If the device has more than 150 short-timeout neighbors and this function is configured or deleted when the CPU usage is high, the neighbors may be intermittently disconnected. If this function is required, you are advised to perform this step before [configuring an OSPFv3 process](#).
- After this step is performed, short-timeout neighbor enhancement takes effect for both OSPF and OSPFv3.

### Step 3 Run commit

The configuration is committed.

----End

## Enabling OSPFv3

Creating an OSPFv3 process is a prerequisite for configuring OSPFv3 features. When creating an OSPFv3 process, you need to manually specify a router ID for it.

## Context

OSPFv3 supports multi-process. Multiple OSPFv3 processes running on one device are differentiated by process IDs. An OSPFv3 process ID is set when an OSPFv3 process is created. The process ID is only locally valid and does not affect packet exchange with other devices.

A router ID is a 32-bit unsigned integer in the format of an IPv4 address. It uniquely identifies a device in an AS. The OSPFv3 router ID must be manually set, and OSPFv3 cannot run properly without a router ID.

When configuring the router ID, ensure that the router ID is unique in an AS. If multiple OSPFv3 processes run on the same device, you are advised to specify different router IDs for the processes.

To ensure OSPFv3 stability, plan router IDs properly during network design and manually set the router IDs during network deployment.

Perform the following steps on each device that needs to run OSPFv3:

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ] [ vpn-instance vpnnname ]**

OSPFv3 is enabled, and the OSPFv3 view is displayed.

You can run the **description** command to configure a description for an OSPFv3 process for easier identification.

### Step 3 Run **router-id router-id**

A router ID is set.

When configuring the router ID, ensure that the router ID is unique in an AS. You can select the IP address of an interface as the router ID.

#### NOTE

Each router ID in an OSPFv3 process must be unique on the entire network. Otherwise, routers cannot establish OSPFv3 neighbor relationships, and the routing information is incorrect.

If a router ID conflict occurs, perform either of the following operations:

- Configure a new router ID.
- Run the **undo ospfv3 router-id auto-recover disable** command to enable the router ID automatic recovery function. After the function is enabled, the system automatically allocates a new router ID.

 NOTE

- If the automatic recovery function is enabled and a router ID conflict occurs between indirectly connected routers in one OSPFv3 area, the system replaces the conflicted router ID with a newly calculated one. The automatic recovery function takes effect on both configured and automatically generated router IDs.
- The system can replace a router ID in a maximum of three attempts in case the router ID conflict persists.

**Step 4 Run commit**

The configuration is committed.

----End

## Enabling OSPFv3 on an Interface

For an interface with multiple instances, you need to specify which instance on the interface is enabled in the OSPFv3 process when enabling OSPFv3 on the interface.

### Context

After enabling OSPFv3 in the system view, you need to enable OSPFv3 on the interface.

Because an interface may have multiple instances, you need to specify which instance of the interface is enabled in the OSPFv3 process when OSPFv3 is enabled on the interface. If no instance ID is specified, the value defaults to 0. The same instance must be enabled on the interfaces between which the neighbor relationship is set up.

Perform the following steps on the router that runs OSPFv3.

## Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run `interface interface-type interface-number`**

The interface view is displayed.

**Step 3 Run `ospfv3 process-id area area-id [ instance instance-id ]`**

OSPFv3 is enabled on the interface.

The specified area ID can be a decimal integer or in the IPv4 address format. Regardless of the specified format, the area ID is displayed as an IPv4 address.

**Step 4 (Optional) Run `ospfv3 network-type { broadcast | nbma | p2mp [ non-broadcast ] | p2p } [ instance instance-id ]`**

A network type is configured for the interface.

When an interface supports multi-instance, specify `instance-id` when enabling OSPFv3 on the interface. If `instance-id` is not specified, the default value 0 is

adopted. In this case, the configured network type of the interface is different from the actual network type. This step is mandatory in such a case.

**Step 5 Run commit**

The configuration is committed.

----End

## Creating an OSPFv3 Area

After an AS is divided into different areas and OSPFv3 interfaces and areas to which these interfaces belong are specified, OSPFv3 can discover and calculate routes in the AS.

## Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run ospfv3 [ *process-id* ]**

The OSPFv3 view is displayed.

**Step 3 Run area *area-id***

The OSPFv3 area view is displayed.

The area ID can be a decimal integer or in the IPv4 address format, but it is displayed in the IPv4 address format.

The description of an OSPFv3 area helps identify special processes by the **description** command.

**Step 4 Run commit**

The configuration is committed.

----End

## (Optional) Configuring the router to Comply with Route Selection Rules Defined in a Standard Protocol

You can configure the router to comply with the route selection rule defined in RFC 1583 or RFC 5340.

## Context

RFC 5340 and RFC 1583 define route selection rules differently. After enabling OSPFv3 on a device, you can configure the device to comply with route selection rules defined in either standard protocol as required. By default, a device complies with the route selection rules defined in RFC 5340. If you want the device to comply with the other protocol, you need to configure the device to comply with the rules defined in RFC 1583. Such configurations ensure that all OSPFv3-enabled devices in an AS comply with the same route selection rules defined in the same standard protocol.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ] [ vpn-instance vpnnname ]**

OSPFv3 is enabled, and the OSPFv3 view is displayed.

### Step 3 Run **rfc1583 compatible**

The router is configured to comply with the route selection rule defined in RFC 1583, rather than RFC 5340.

### Step 4 Run **commit**

The configuration is committed.

----End

## (Optional) Configuring an Alarm Threshold for the Number of LSAs Learned by OSPFv3

You can configure an alarm threshold for the number of LSAs learned by OSPFv3 so that an alarm is reported when this threshold is reached or exceeded.

## Context

If the number of external routes that OSPFv3 imports and advertises exceeds the routing table capacity of a device, the device may restart unexpectedly. To prevent this problem and ensure that the device runs properly, you can set an alarm threshold for the number of LSAs learned by OSPFv3 and enable overload control.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **maximum received-lsa threshold value [ overload-limit ]**

An alarm threshold is configured for the number of LSAs learned by OSPFv3. You can determine whether to enable overload control as required.

- If an alarm threshold is configured but overload control is not, only an alarm is reported when this threshold is reached or exceeded.
- If both an alarm threshold and overload control are configured, an alarm is reported when this threshold is reached or exceeded. In addition, OSPFv3 no longer learns new LSAs when the following conditions are met:
  - The number of LSAs learned by OSPFv3 reaches or exceeds the alarm threshold.
  - The memory is in the danger state, and the memory used by the OSPFv3 LSDB component ranks in the top 3.

- The **ospf memory-overload control** command configuration exists (The corresponding function is enabled by default).

#### Step 4 Run **commit**

The configuration is committed.

----End

### (Optional) Configuring the Maximum Number of Packet Retransmission Attempts

When no response to DD packets, Update packets, or Request packets is received, the retransmission mechanism is used and the maximum number of packet retransmission attempts is set to prevent dead loops caused by repeated transmissions.

#### Context

If no response is received when the maximum number of packet retransmission attempts is reached, the neighbor relationship will be interrupted.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

##### Step 3 Run **retransmission-limit [ max-number ]**

The maximum number of OSPFv3 packet retransmission attempts is set.

##### Step 4 Run **commit**

The configuration is committed.

----End

### (Optional) Disabling an Interface from Receiving and Sending OSPFv3 Packets

Disabling an interface from receiving and sending OSPFv3 packets prevents other network devices from obtaining OSPFv3 routing information and prevents the local device from receiving routing updates advertised by other network devices.

#### Context

To prevent a device interface from sending its OSPFv3 routing information to other devices on the network and disable the device interface from receiving routing updates from other devices, you can suppress the interface from sending and receiving OSPFv3 packets. After an interface is disabled from receiving and sending OSPFv3 packets, the interface can still advertise its routes but cannot exchange Hello packets with others. Therefore, no neighbor relationship can be established between the interface and others. This can enhance the networking adaptability of OSPFv3 and reduce system resource consumption.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run ospfv3 [ *process-id* ]

The OSPFv3 process view is displayed.

### Step 3 Run silent-interface *interface-type interface-number*

The interface is disabled from receiving and sending OSPFv3 packets.

#### NOTE

In different processes, you can suppress the same interface from sending and receiving OSPFv3 packets. However, the **silent-interface** configuration takes effect only in the specified process of an interface, not all processes associated with the interface.

### Step 4 Run commit

The configuration is committed.

----End

## Verifying the Basic OSPFv3 Configuration

After configuring basic OSPFv3 functions, verify information about neighbors, interfaces, and the OSPFv3 routing table.

## Prerequisites

Basic OSPFv3 functions have been configured.

## Procedure

- Run the **display ospfv3 [ *process-id* ] [ area *area-id* ] peer [ *interface-type interface-number* ] [ verbose ]** command in any view to view information about OSPFv3 neighbors.
- Run the **display ospfv3 [ *process-id* ] [ area *area-id* ] [ *interface-type interface-number* ]** command in any view to view information about an OSPFv3 interface.
- Run the **display ospfv3 [ *process-id* ] routing** command in any view to view information about an OSPFv3 routing table.
- Run the **display ospfv3 [ *process-id* ] cumulative** command in any view to view OSPFv3 statistics.
- Run the **display ospfv3 [ *process-id* ] error [ lsa | interface *interface-type interface-number* ]** command in any view to view OSPFv3 errors.
- Run the **display ospfv3 [ *process-id* ] next-hop** command in any view to view the OSPFv3 next-hop routing table.
- Run the **display ospfv3 [ *process-id* ] request-list [ statistics | [ area *area-id* | peer *router-id* | interface *interface-type interface-number* ] \* ]** command in any view to view the statistics of the request list of OSPFv3.

- Run the **display ospfv3 [ process-id ] retrans-list [ statistics | { area area-id } peer router-id | interface interface-type interface-number } \* ]** command in any view to view the statistics of the OSPFv3 retransmission list.
- Run the **display ospfv3 [ process-id ] spf-statistics [ verbose ]** command in any view to view OSPFv3 route calculation statistics.
- Run the **display ospfv3 [ process-id ] statistics updated-lsa [ originate-router advertising-router-id | history ]** command in any view to view the frequent updates of the Link State Advertisements (LSAs) that the Link-state Database (LSDB) receives.
- Run the **display ospfv3 [ process-id ] topology [ area area-id ] [ statistics | verbose ]** command in any view to view information about the topology in an OSPFv3 area.

----End

#### 1.1.5.2.4 Configuring OSPFv3 Attributes on Different Types of Networks

By setting network types for OSPFv3 interfaces and adjusting OSPFv3 attributes, you can build OSPFv3 networks flexibly.

#### Usage Scenario

Based on the types of link layer protocols, OSPFv3 classifies networks into the following types:

- P2MP: Because P2MP is not a link layer protocol, each P2MP network is changed from a network of another type.
- NBMA: If the link layer protocol is X.25, OSPFv3 defaults the network type to NBMA.
- Broadcast: If the link layer protocol is GigabitEthernet or FDDI, OSPFv3 defaults the network type to broadcast.
- P2P: If the link layer protocol is PPP, LAPB, OSPFv3 defaults the network type to P2P.

You can change network types and configure OSPFv3 features to flexibly build networks without changing link layer protocols.

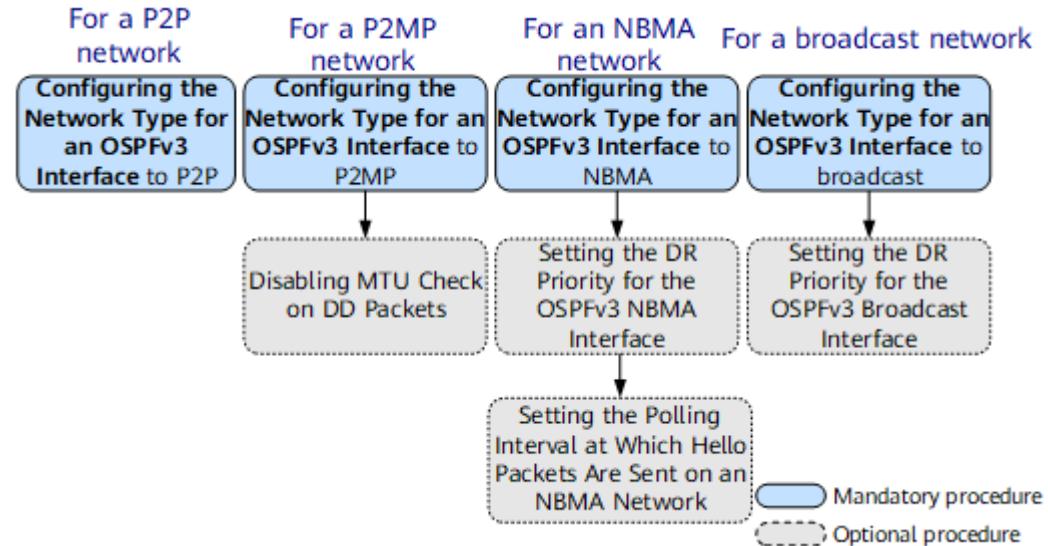
#### Pre-configuration Tasks

Before configuring OSPFv3 attributes on different types of networks, complete the following tasks:

- Configure a link layer protocol.
- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

## Configuration Procedures

Figure 1-184 Flowchart for configuring OSPFv3 attributes on different types of networks



## Configuring the Network Type for an Interface

OSPFv3 classifies networks into broadcast, P2P, P2MP, and NBMA networks based on link layer protocols. By configuring network types for interfaces, you can change the network types of interfaces.

### Context

By default, the network type of an interface is determined by the physical interface. The network type of Ethernet interfaces is **broadcast**.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 3 Run **ospfv3 network-type { broadcast | nbma | p2mp [ non-broadcast ] | p2p } [ instance instance-id ]**

The network type is configured for the OSPFv3 interface.

After a new network type is configured for an interface, the network type of the original interface is replaced.

#### NOTE

The network types of two OSPFv3 interfaces at both ends of a link must be identical. Otherwise, the OSPFv3 neighbor relationship cannot be established.

#### Step 4 Run commit

The configuration is committed.

----End

### (Optional) Setting the DR Priority for the OSPFv3 Broadcast or NBMA Interface

You can specify the Designated Router (DR) priority for each interface on a broadcast or a non-broadcast multiple access (NBMA) network for DR/Backup Designated Router (BDR) election on the network.

### Context

When configuring broadcast networks or NBMA networks, you can specify the DR priority for each interface for DR/BDR election on the network. The greater the value, the higher the priority.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run **interface** *interface-type interface-number*

The interface view is displayed.

#### Step 3 Run **ospfv3 dr-priority** *priority*[ **instance** *instance-id* ]

The DR priority of the OSPFv3 interface is set.

#### Step 4 (Optional) Run **ospfv3 timer wait** *interval* [ **instance** *instance-id* ]

The wait timer on an OSPFv3 interface is set.

If no Backup Seen event is received within the *interval*, the designated router (DR) election starts. Setting a proper value for the wait timer can slow down changes of the DR and the backup designated router (BDR) on the network, reducing network flapping. When setting the wait timer, note the following points:

- The wait timer takes effect only on broadcast and NBMA interfaces.
- The value of the wait timer cannot be greater than the value of the dead timer.

#### Step 5 Run commit

The configuration is committed.

----End

### Follow-up Procedure

#### NOTICE

Changing the priority leads to a DR/BDR re-election and interrupts the OSPFv3 adjacency between routers. Therefore, changing the priority is not recommended.

Reconfiguring the DR priority for an interface does not change the DR or BDR on the network. To reelect a DR or BDR, perform either of the following operations.

- Restart the OSPFv3 processes on all the routers.
- Run the **shutdown** and then **undo shutdown** commands on the interfaces where neighbor relationships are established.

## (Optional) Setting the Polling Interval at Which Hello Packets Are Sent on an NBMA Network

On an NBMA network, a device sends Hello packets to a neighbor that is Down at a polling interval.

### Context

Perform the following steps on the router running OSPFv3.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The OSPFv3 interface view is displayed.

#### Step 3 Run **ospfv3 timer poll interval [ instance instance-id ]**

The polling interval at which Hello packets are sent is set on the NBMA interface.

The parameter **poll interval** specifies the polling interval at which Hello packets are sent.

#### Step 4 Run **commit**

The configuration is committed.

----End

## (Optional) Disabling MTU Check on DD Packets

After you prevent an interface from checking the Maximum Transmission Unit (MTU) field in received Database Description (DD) packets, the device can receive packets with the MTU field as 0.

### Context

Perform the following steps on the router that runs OSPFv3:

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

**Step 3 Run `ospfv3 mtu-ignore [ instance instance-id ]`**

The MTU check on DD packets is disabled.

After the command is used, the interface does not check the MTU field of received DD packets.

**Step 4 Run `commit`**

The configuration is committed.

----End

## Verifying the Configuration of OSPFv3 Attributes in Different Types of Networks

After configuring attributes of OSPFv3 interfaces on different types of networks, verify information about OSPFv3 interfaces.

### Prerequisites

OSPFv3 interface attributes on different types of networks have been configured.

### Procedure

**Step 1 Run the `display ospfv3 [ process-id ] interface [ no-peer | area area-id ] [ interface-type interface-number ]` command in any view to view information about an OSPFv3 interface.**

----End

### 1.1.5.2.5 Adjusting OSPFv3 Route Selection

By adjusting OSPFv3 route selection, you can enable OSPF to meet the requirements of complex networks.

### Usage Scenario

To meet the requirements of complex networks, you can adjust OSPFv3 route selection rule by configuring the following OSPFv3 route attributes:

- Cost on the OSPFv3 interface
- Load balancing among equal-cost routes

### Pre-configuration Tasks

Before adjusting OSPFv3 route selection, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

### Setting the Link Cost on an OSPFv3 Interface

OSPFv3 can automatically calculate the link cost for an interface based on the interface bandwidth. You can also set the link cost for the interface.

## Context

You can control the route cost by setting different link costs for OSPFv3 interfaces.

Perform the following steps on the router that runs OSPFv3:

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **ospfv3 cost value [ instance instanceId ]**

The link cost is set on the OSPFv3 interface.

If you do not set the cost of an OSPFv3 interface using the **ospfv3 cost cost** command, OSPFv3 automatically calculates the cost for the interface based on the interface bandwidth. The calculation formula is as follows: Cost of the interface = Bandwidth reference value/Interface bandwidth. The integer of the calculated result is the cost of the interface. If the calculated result is smaller than 1, the cost value is 1. Changing the bandwidth reference value can change the cost of an interface.

Perform the following steps to change the bandwidth reference value:

#### 1. Run **system-view**

The system view is displayed.

#### 2. Run **ospfv3 [ process-id ]**

OSPFv3 is enabled, and the OSPFv3 view is displayed.

#### 3. Run **bandwidth-reference value**

The bandwidth reference value is set.

#### 4. Run **commit**

The configuration is committed.

### Step 4 Run **commit**

The configuration is committed.

----End

## Setting the Maximum Number of Equal-Cost Routes

Routes of the same routing protocol with the same destination and cost are called equal-cost routes, and traffic can be load-balanced among these routes.

## Context

Perform the following steps on the router that runs OSPFv3:

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **maximum load-balancing number**

The maximum number of equal-cost routes is set.

If the number of equal-cost routes is greater than *number* specified in the **maximum load-balancing** command, routes are selected for load balancing based on the following criteria:

1. Route priority: Routes with smaller priority values are selected for load balancing. For details about route priority configuration, see [Step 4](#).
2. Interface index: If routes have the same priority, those with greater interface index values are selected for load balancing.
3. Next hop IP address: If routes have the same priority and interface index, those with larger IP addresses are selected for load balancing.

### Step 4 (Optional) Run **nexthop neighbor-id { interface-name | interfaceType interfaceNum } weight value**

The preference for equal-cost routes is set.

OSPFv3 selects a next hop from these equal-cost routes according to the weight. The smaller the weight is, the higher the route preference is.

### Step 5 Run **commit**

The configuration is committed.

----End

## Setting the convergence priority for OSPFv3 routes

You can adjust and optimize route selection by setting the convergence priority for OSPFv3 routes.

## Context

You can set a convergence priority for the OSPFv3 routes matching the specified IPv6 prefix list. The configuration takes effect on the public network only.

OSPFv3 route calculation, LSA flooding, and LSDB synchronization can be implemented according to the configured priority. Therefore, route convergence can be controlled.

When an LSA meets multiple priorities, the highest priority takes effect.

OSPFv3 calculates LSAs in the sequence of intra-area routes, inter-area routes, and AS external routes. This command enables OSPFv3 to calculate the three types of routes separately according to the specified route calculation priorities. The convergence priority sequence is as follows: critical > high > medium > low. To

speed up the processing of LSAs with the higher priority, during LSA flooding, the LSAs need to be placed into the corresponding critical, high, medium, and low queues according to priorities.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ip ipv6-prefix *ipv6-prefix-name* [ index *index-number* ] matchMode *ipv6-address masklen* [ match-network ] [ greater-equal *greater-equal-value* ] [ less-equal *less-equal-value* ]**

An IPv6 prefix list is configured.

### Step 3 Run **ospfv3 [ *process-id* ]**

The OSPFv3 view is displayed.

### Step 4 Run **prefix-priority { critical | high | medium | very-low } ipv6-prefix *ipv6-prefix-name***

The convergence priority for OSPFv3 routes is set.

### Step 5 Run **commit**

The configuration is committed.

----End

## Configuring an OSPFv3 Interface to Automatically Adjust the Link Cost

Configuring an OSPFv3 interface to automatically adjust the link cost based on link quality facilitates route selection control and improves network reliability.

## Context

A bit error refers to the deviation between a bit that is sent and the bit that is received. The bit error rate (BER) refers to the number of bit errors divided by the total number of bits transferred during a studied time interval. During data transmission, a high BER will degrade or even interrupt services in extreme cases.

To prevent this problem, configure OSPFv3 interfaces to automatically adjust link costs based on link quality so that unreliable links are not used by the optimal routes.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface *interface-type* *interface-number***

The interface view is displayed.

### Step 3 Run **ipv6 enable**

IPv6 is enabled on the interface.

**Step 4** Run **ospfv3 process-id area area-id [ instance instance-id ]**

OSPFv3 is enabled on the interface.

**Step 5** Run **ospfv3 link-quality low incr-cost { cost | max-reachable } [ instance instance-id ]**

The OSPFv3 interface is enabled to automatically adjust the link cost based on link quality.

 **NOTE**

The *cost* parameter specifies the link cost adjustment value. After this parameter is specified:

- If the link quality changes from good to low, the link cost equals the original link cost plus the adjustment value. The maximum link cost allowed is 65535.
- If the link quality changes from low to good, the original link cost applies.

**Step 6** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of OSPFv3 Route Selection

After setting OSPF route attributes, verify information about OSPFv3 interfaces and the routing table.

## Prerequisites

OSPFv3 route attributes have been configured.

## Procedure

- Run the **display ospfv3 [ process-id ] interface [ no-peer | area area-id ] [ interface-type interface-number ]** command in any view to view information about an OSPFv3 interface.
- Run the **display ospfv3 [ process-id ] routing** command in any view to view information about an OSPFv3 routing table.
- Run the **display ospfv3 [ process-id ] ecmp-group** command in any view to view information about OSPFv3 ECMP groups.

----End

### 1.1.5.2.6 Controlling OSPFv3 Routing Information

This section describes how to control OSPFv3 routing information.

## Usage Scenario

Through the configuration in this section, you can control the advertising and receiving of OSPFv3 routing information and configure OSPFv3 to import external routes.

You can control the following types of routes:

- Routes outside the local OSPFv3 area

This function can be configured on any router that runs OSPFv3. For example, you can enable the router to filter received routes and set the maximum number of equal-cost routes.

- Routes inside the local OSPFv3 area

This function can be configured only on ABRs. For example, when an ABR has multiple routes to a destination in the local area, you can configure the ABR to summarize the routes and send only a summary LSA to the backbone area.

## Pre-configuration Tasks

Before controlling OSPFv3 routing information, complete the following tasks:

- Enable IPv6.
- [Configure basic OSPFv3 functions](#).

## Configuring OSPFv3 to Import External Routes

Importing the routes discovered by other routing protocols can enrich OSPFv3 routing information.

## Context

Because OSPFv3 is a link-state routing protocol and cannot filter LSAs to be advertised directly, it can only filter routes when importing them. As such, only the routes that match the filtering conditions can be converted into LSAs and advertised.

### NOTICE

OSPFv3 and other dynamic routing protocols such as IS-IS and BGP often import routes from each other. If no routing policy is configured or a routing policy is incorrectly configured on a device where IS-IS, OSPFv3, and BGP import routes from each other, a Layer 3 routing loop may occur due to a route selection result change. As a result, services are compromised. For details about the cause of the loop, see [Routing Loop Detection for Routes Imported to OSPFv3](#).

## Procedure

### Step 1 Run `system-view`

The system view is displayed.

### Step 2 Run `ospfv3 [ process-id ]`

The OSPFv3 view is displayed.

### Step 3 Run `default { cost costvalue | tag tagvalue | type typevalue } *`

The default cost is set for imported routes.

**Step 4** Run **import-route { bgp [ permit-ibgp ] | direct | static | unr } [ { cost *cost* | inherit-cost } | tag *tag* | type *type* | { route-policy *route-policy-name* | route-filter *route-filter-name* } ] \***

The device is configured to import external routes.

 **NOTE**

The **import-route** command cannot be used to import the default route from another AS.

**Step 5** (Optional) Run **import-route limit *limit-number* [ threshold-alarm { upper-limit *upper-limit-value* | lower-limit *lower-limit-value* } ] \***

A limit is configured on the number of imported external routes to be advertised by OSPFv3.

If OSPFv3 imports a large number of external routes and advertises them to a device with a small routing table capacity, the device may restart unexpectedly. To prevent this problem, configure a limit on the number of imported external routes to be advertised by OSPFv3.

Ensure that *upper-limit-value* is greater than or equal to *lower-limit-value*.

**Step 6** Run **commit**

The configuration is committed.

----End

## Configuring OSPFv3 to Advertise Default Routes to OSPFv3 Areas

You can configure OSPFv3 to advertise default routes to OSPFv3 areas. Only the routes that meet the conditions can be converted into LSAs and advertised.

### Context

Perform the following steps on the router that runs OSPFv3:

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

**Step 3** Run **default-route-advertise [ always | cost *cost* | type *type* | tag *tag* | { route-policy *route-policy-name* | route-filter *route-filter-name* } | permit-preference-less-than *preference-value* ] \***

The device is configured to advertise default routes to OSPFv3 areas.

 **NOTE**

To prevent loops, you are advised to specify **permit-preference-less-than** to prevent low-priority active default routes from being imported. This parameter is used only when **always** is not specified.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring OSPFv3 to Filter Received Routes

After receiving Link State Advertisements (LSAs), OSPFv3 determines whether to add the calculated routes to the local routing table based on a filtering policy.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run ospfv3 [ process-id ]

The OSPFv3 view is displayed.

#### Step 3 Run any of the following commands as required:

- Configure a basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

- c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \***

A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- d. Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

- e. Run **filter-policy { acl6-number| acl6-name acl6-name } import**

An import policy that is based on the ACL is configured to filter received routes.

- Based on an IPv6 prefix list:

Run **filter-policy ipv6-prefix ipv6-prefix-name import**

An import policy that is based on an IPv6 prefix list is configured to filter received routes.

- Based on a route-policy:

Run **filter-policy route-policy route-policy-name import**

An import policy that is based on a route-policy is configured to filter received routes.

- Based on a route-filter:

Run **filter-policy route-filter route-filter-name import**

An import policy that is based on a route-filter is configured to filter received routes.

The **filter-policy import** command is used to filter the routes calculated by OSPFv3. Only the routes that match the filtering rules are added to the routing table and can be advertised. Routes that do not match the filtering rules can be added to the OSPFv3 routing table but not to the routing information base (RIB) and cannot be advertised. Therefore, the LSDB is not affected regardless of whether the received routes match the filtering rules.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring OSPFv3 to Filter the Routes to Be Advertised

After filtering conditions are set for imported routes, only the routes that pass the filtering can be advertised.

## Context

When OSPFv3 receives LSAs, it can filter the received routes based on a filtering policy before advertising them to neighbors.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 (Optional) Run **default-route-advertise [ [ always | permit-calculate-other | cost cost | type type | tag tag | distribute-delay delay | route-policy route-policy-name | route-filter route-filter-name } | permit-preference-less-than preference-val ] ] \***

The device is configured to advertise default routes to OSPFv3 areas.



To prevent loops, you are advised to specify **permit-preference-less-than** to prevent low-priority active default routes from being imported. This parameter is used only when **always** is not specified.

### Step 4 Run any of the following commands as required:

- Configure a basic ACL:

#### a. Run **quit**

Return to the system view.

#### b. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

#### c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \***

A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.

- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:
  - If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.  
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.
- d. Run **ospfv3 [ process-id ]**  
The OSPFv3 view is displayed.
- e. Run **filter-policy { acl-number | acl6-name acl6-name } export [ bgp | direct | static | unr | { isis | ospfv3 | ripng } [ process-id ] ]**  
An export policy is configured to filter the routes that are imported using the **import-route** command and are to be advertised. Only the matched routes can be advertised.
- Based on an IPv6 prefix list:  
Run **filter-policy ipv6-prefix ipv6-prefix-name export [ bgp | direct | static | unr | { isis | ospfv3 | ripng } [ process-id ] ]**  
An export policy is configured to filter the routes that are imported using the **import-route** command and are to be advertised. Only the matched routes can be advertised.

You can specify the protocol type or *process-id* to match the routes of a specified protocol or process. If no protocol type or *process-id* is specified, OSPFv3 filters all imported routes.

#### Step 5 Run commit

The configuration is committed.

----End

### Configuring OSPFv3 to Filter LSAs in an Area

Filtering LSAs in an area can prevent unnecessary LSA transmission. This reduces the size of the LSDB on the neighboring device and speeds up network convergence.

### Context

After filtering conditions are set for the incoming or outgoing Type 3 LSAs (Inter-Area-Prefix LSAs) in an area, only the Type 3 LSAs that meet the filtering conditions can be received or advertised.

This function is applicable only to the ABR.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **area area-id**

The OSPFv3 area view is displayed.

### Step 4 Filter incoming or outgoing Type 3 LSAs in the area.

- Filter incoming Type 3 LSAs in the area. Run any of the following commands as required:

- Configure a basic ACL:

- i. Run **quit**

Return to the OSPFv3 view.

- ii. Run **quit**

Return to the system view.

- iii. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

- iv. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \***

A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

v. Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

vi. Run **area area-id**

The OSPFv3 area view is displayed.

vii. Run **filter { acl6-number | acl6-name acl6-name } import**

Incoming Type 3 LSAs in the area are filtered based on the ACL.

- Based on an IPv6 prefix list:

Run **filter ipv6-prefix ipv6-prefix-name import**

Incoming Type 3 LSAs in the area are filtered based on an IP prefix list.

- Based on a route-policy:

Run **filter route-policy route-policy-name import**

Incoming Type 3 LSAs in the area are filtered based on a route-policy.

- Based on a route-filter:

Run **filter route-filter route-filter-name import**

Incoming Type 3 LSAs in the area are filtered based on a route-filter.

- Filter outgoing Type 3 LSAs in the area. Run any of the following commands as required:

- Configure a basic ACL:

i. Run **quit**

Return to the OSPFv3 view.

ii. Run **quit**

Return to the system view.

iii. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

iv. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \***

A rule is configured for the ACL. When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
  - If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
  - If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
  - If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
  - Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.  
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.
- v. Run **ospfv3 [ process-id ]**  
The OSPFv3 view is displayed.
  - vi. Run **area area-id**  
The OSPFv3 area view is displayed.
  - vii. Run **filter { acl6-number | acl6-name acl6-name } export**  
Outgoing Type 3 LSAs in the area are filtered based on the ACL.
    - Based on an IPv6 prefix list:  
Run **filter ipv6-prefix ipv6-prefix-name export**  
Outgoing Type 3 LSAs in the area are filtered based on an IP prefix list.
    - Based on a route-policy:  
Run **filter route-policy route-policy-name export**  
Outgoing Type 3 LSAs in the area are filtered based on a route-policy.
    - Based on a route-filter:  
Run **filter route-filter route-filter-name export**  
Outgoing Type 3 LSAs in the area are filtered based on a route-filter.

#### Step 5 Run commit

The configuration is committed.

----End

#### (Optional) Configuring OSPFv3 to Discard Specified LSAs

You can configure a device to discard specified LSAs in an OSPFv3 process.

## Context

OSPFv3 can be configured to discard specified LSAs in the following scenarios:

- When devices on the entire network restart repeatedly due to abnormal LSAs and you have located the LSA that causes protocol restarts, you can configure this function as a last resort to prevent the device from restarting continuously. However, if this function is incorrectly configured, routing loops may occur.
- If an LSA is identified as an attack packet, which is not supposed to appear in the local area, and has caused serious problems, such as device restarts, you can configure this function to filter out the LSA under the condition that the attack source cannot be located temporarily and that the LSA does not affect topology path computation.
- If an LSA is identified as an attack packet, which is not supposed to appear in the local area, and it affects topology path computation and has caused serious problems, such as network-wide device restarts, you can configure this function on each device to discard the LSA to prevent it from participating in network-wide calculation.

### NOTE

To filter out the LSA that affects topology path computation, you must ensure that it is removed from all the LSDBs on the entire network. Otherwise, routing loops may occur.

- If an LSA is identified as an unreachable residual LSA and the device that advertised the LSA becomes permanently unreachable, you can configure this function to filter out the LSA upon reception under the condition that the LSA does not affect topology path computation.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **ignore-receive-lsa advertise-router adv-rtr-id [ lsa-type type-value [ area { area-id | area-idipv4 } ] | link-state-id ls-id ] \***

The device is configured to discard LSAs of a specified type.

 NOTE

- You are not advised to run this command to filter out the LSAs that exist on the network as running this command may filter out normal LSAs.
- This command cannot be used to defend against attacks as it goes against protocol processing rules and affects services. Therefore, exercise caution when running this command.
- As an attack LSA can have any key, it is difficult to defend against the LSA using this command. Therefore, you are advised to directly isolate the attack source.
- If this command is incorrectly configured, services cannot be restored even if the **undo ignore-receive-lsa advertise-router adv-rtr-id [ lsa-type type-value [ area { area-id | area-id|ipv4 } ] | link-state-id ls-id ] \* command** is run. In this case, you may need to reset the process or neighbor to restore services.
- If the fault is caused by a bug, you are advised to run this command temporarily. After the patch is installed, run the **undo ignore-receive-lsa advertise-router adv-rtr-id [ lsa-type type-value [ area { area-id | area-id|ipv4 } ] | link-state-id ls-id ] \* command** immediately and check whether services are affected. If services are affected, re-establish all neighbor relationships to restore services.

**Step 4 Run commit**

The configuration is committed.

----End

## Configuring OSPFv3 Route Summarization

Routes with the same IPv6 prefix can be summarized into one route. On a large-scale OSPFv3 network, route lookup may slow down because of the large size of the routing table. To reduce the routing table size and simplify management, configure route summarization. With route summarization, if a link connected to a device within an IPv6 address range that has been summarized alternates between Up and Down, the link status change is not advertised to the devices beyond the IPv6 address range. This prevents route flapping and improves network stability.

## Context

OSPFv3 route summarization is classified as follows:

- Route summarization on an ABR, which is configured using the **abr-summary** command. When an ABR transmits routing information to other areas, it generates Type 3 LSAs for each network segment. If contiguous segments exist in this area, run the **abr-summary** command to summarize these segments into one segment so that the ABR sends only one summarized LSA.
- Route summarization on an ASBR, which is configured using the **asbr-summary** command. The ASBR summarizes routes of imported Type 5 LSAs within the summarized address range. If a large number of routes with the same IPv6 prefix are imported, run the **asbr-summary** command to summarize them into one route.

## Procedure

- Configure route summarization on an ABR.
  - a. Run **system-view**

- The system view is displayed.
- b. Run **ospfv3** [ *process-id* ]
- The OSPFv3 view is displayed.
- c. Run **area** *area-id*
- The OSPFv3 area view is displayed.
- d. Run **abr-summary** *ipv6-address prefix-length* [ **not-advertise** | **cost** *cost-value* | **hold-max-cost** *interval* ] \* The ABR is configured to summarize intra-area routes.
- The **not-advertise** parameter indicates that the routes in this network segment are not advertised.
  - The **cost** *cost-value* parameter specifies a cost for the summary route.
  - The **hold-max-cost** *interval* parameter specifies the period during which the maximum cost is advertised in the summary route.
- e. Run **commit**
- The configuration is committed.
- Configure route summarization on an ASBR.
- a. Run **system-view**
- The system view is displayed.
- b. Run **ospfv3** [ *process-id* ]
- The OSPFv3 view is displayed.
- c. Run **asbr-summary** *ipv6-address prefix-length* [ **not-advertise** | **tag** *tag-value* | **cost** *cost-value* | **distribute-delay** *interval* ] \*
- The ASBR is configured to summarize the routes imported by OSPFv3.
- The **cost** *cost-value* parameter specifies a cost for the summary route.
  - The **tag** *tag-value* parameter specifies a tag value used to control summary route advertisement.
  - The **not-advertise** parameter indicates that the summary IPv6 routes that match the specified IPv6 prefix or prefix length are not advertised.
  - The **distribute-delay** *interval* parameter sets a delay for advertising a summary route. This ensures that the summary route advertised each time contains more valid routes and prevents network flapping caused by incorrect routing information.
- d. Run **commit**
- The configuration is committed.

----End

## Verifying the Configuration of OSPFv3 Routing Information Control

After controlling OSPFv3 routing information, verify the OSPFv3 LSDB.

### Prerequisites

OSPFv3 routing information has been controlled.

### Procedure

- Run the **display ospfv3 [ process-id ] lsdb** command to view the OSPFv3 LSDB.
- Run either of the following commands to check OSPFv3 route information:
  - Run the **display ospfv3 [ process-id ] abr-summary-list [ ipv6-address prefix-length ]** command to check information about the summarized routes on an ABR.
  - Run the **display ospfv3 [ process-id ] asbr-summary [ ipv6-address prefix-length ] [ verbose ]** command to check information about the summarized routes on an ASBR.

----End

### 1.1.5.2.7 Configuring OSPFv3 Delay Advertisement

With OSPFv3 delay advertisement, intra-domain link delay information can be collected and flooded.

### Usage Scenario

Traditional path computation algorithms compute the optimal path to a destination address based on costs. However, the path computed in this way may not have the minimum delay. For services that have stringent requirements on delay, path computation can be performed based on the delay rather than on the cost to ensure that service traffic is transmitted along the path with the minimum delay. After OSPFv3 delay advertisement is configured, OSPFv3 collects and floods intra-domain link delay information and reports the information to the controller through BGP-LS. The controller then computes paths on the P2P network based on delay constraints.

### Pre-configuration Tasks

Before configuring delay advertisement, configure TWAMP Light to detect delay information. The configuration is as follows:

- Configure the TWAMP Light controller function on the local end.
  - a. Configure the TWAMP Light client function, and create a test session.
    - i. Run **system-view**  
The system view is displayed.
    - ii. Run **nqa twamp-light**  
The TWAMP Light view is displayed.
    - iii. Run **client**

- The TWAMP Light client function is enabled, and the TWAMP Light client view is displayed.
- iv. Run **test-session session-id sender-ipv6 sender-address-v6 reflector-ipv6 reflector-address-v6 sender-port sender-port reflector-port reflector-port [ dscp dscp-value | padding padding-length | padding-type padding-type | description description ] \***  
A test session is created on the initiator.
  - v. Run **quit**  
Return to the TWAMP Light view.
- b. Configure the TWAMP Light Sender and start the TWAMP Light performance test.
    - i. Run **sender**  
The TWAMP Light Sender function is enabled, and the TWAMP Light Sender view is displayed.
    - ii. Run **test start-continual test-session session-id [ period { 10 | 100 | 1000 | 30000 } ] [ time-out time-out ]**  
Continual measurement is started.
    - iii. Run **commit**  
The configuration is committed.
- Configure the TWAMP Light Responder on the peer end.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **nqa twamp-light**  
The TWAMP Light view is displayed.
    - c. Run **responder**  
The TWAMP Light Responder function is enabled, and the TWAMP Light Responder view is displayed.
    - d. Run **test-session session-id local-ipv6 local-ipv6-address remote-ipv6 remote-ipv6-address local-port local-port remote-port remote-port interface { interface-type interface-number | interface-name } [ anti-loop-on ] [ description description ]**  
A test session is created on the reflector.
    - e. Run **commit**  
The configuration is committed.

For details about how to configure TWAMP Light, see TWAMP Light.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

An OSPFv3 process is created, and the OSPFv3 view is displayed.

*process-id* specifies an OSPFv3 process. If *process-id* is not specified, the default process ID 1 is used.

**Step 3** Run **router-id** *router-id*

A router ID is configured.

A router ID uniquely identifies an OSPFv3 process in an AS. If no router ID is configured, the OSPFv3 process cannot run.

**Step 4** Run **metric-delay** [ **average** | **variation** ] **advertisement enable**

OSPFv3 delay advertisement is configured.

**Step 5** (Optional) Run **metric-delay normalize interval** *interval-value* [ **offset** *offset-value* ]

The link delay tolerance in the OSPFv3 process is configured.

For the delay-based path computation algorithm, the delay differences of links are different, and the differences may be small. However, even if the delay differences are small, only one optimal path can be generated according to the existing SPF algorithm, and load balancing cannot be implemented within a certain delay tolerance range. As a result, link resources on the network cannot be fully utilized. To resolve this problem to the greatest extent, normalization processing may be performed on the link delays with a small difference or a difference within an acceptable range so that load balancing can be implemented and link resources on the network can be fully utilized.

**Step 6** (Optional) Run **metric-delay suppress timer** *timer-value* **percent-threshold** *percent-value* **absolute-threshold** *absolute-value*

Delay advertisement suppression is configured.

If delay variation occurs frequently, the delay-based route calculation changes frequently. As a result, routing information is flooded and reported repeatedly. In this case, you can configure this function to suppress delay advertisement.

**Step 7** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, run the **display ospfv3 traffic-eng** command to check the delay information advertised by OSPFv3.

### 1.1.5.2.8 Disabling DN Bit Setting in OSPFv3 LSAs

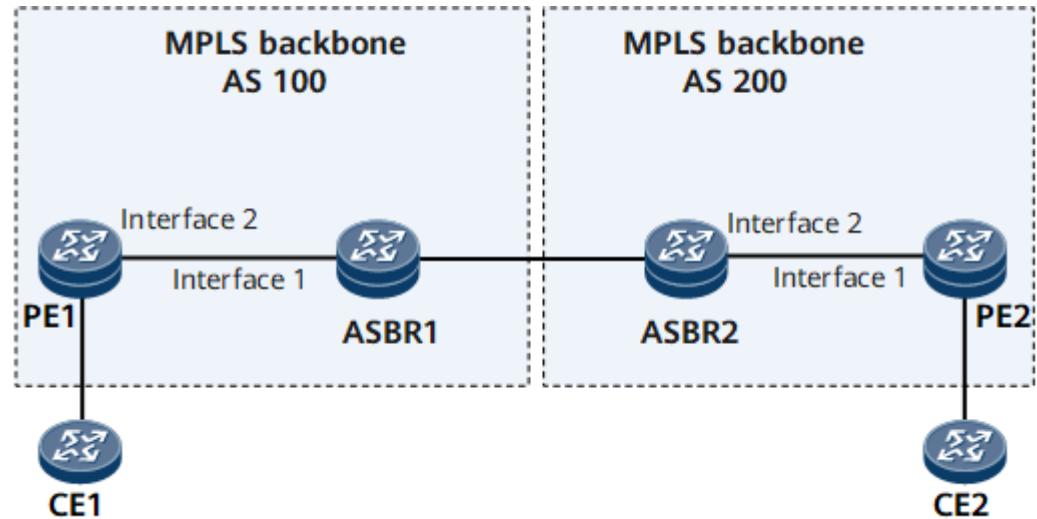
In a VPN Option A scenario, disabling DN bit setting in OSPFv3 LSAs allows PEs in different ASs to communicate with each other.

## Usage Scenario

[Figure 1-185](#) shows a VPN Option A scenario where an OSPFv3 neighbor relationship is established between ASBR1 and ASBR2. Each of the ASBRs imports

BGP routes from the connected PE to the OSPFv3 process, generates LSAs, and advertises the LSAs to the peer ASBR. The ASBRs, however, cannot use the received LSAs to generate OSPFv3 routes. This is because LSAs with the DN bit set cannot be used for route calculation, as defined in a standard protocol. Consequently, no OSPFv3 routes will be imported into BGP processes on the ASBRs, and no BGP routes will be advertised to the connected PEs, causing traffic loss. To prevent this issue, you can disable DN bit setting in OSPFv3 LSAs.

**Figure 1-185** OSPFv3 application in the VPN Option A scenario



## Pre-configuration Tasks

Configure a VPN instance before disabling DN bit setting in OSPFv3 LSAs.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ] [ vpn-instance vpnname ]**

An OSPFv3 process bound to the existing VPN instance is started, and the view of the OSPFv3 process is displayed.

An OSPFv3 process can be bound to only one VPN instance. If an OSPFv3 process is not bound to any VPN instance before being started, this process is a public network process. The public network OSPFv3 process cannot be bound to a VPN instance.

#### NOTE

Deleting a VPN instance or disabling the IPv6 address family in the VPN instance will delete all the associated OSPFv3 processes.

### Step 3 Run **router-id router-id**

A router ID is configured.

A router ID uniquely identifies an OSPFv3 process in an AS. If no router ID is configured, the OSPFv3 process cannot run.

**Step 4** Run **import-route bgp [ cost cost | { route-policy route-policy-name | route-filter route-filter-name } | tag tag | type type ]\***

The device is configured to import BGP routes into OSPFv3.

**Step 5** Run **dn-bit-set disable { summary | ase | nssa }**

DN bit setting in OSPFv3 LSAs is disabled.

**Step 6** Run **commit**

The configuration is committed.

----End

### 1.1.5.2.9 Configuring OSPFv3 Neighbor Relationship Flapping Suppression

OSPFv3 neighbor relationship flapping suppression works by delaying OSPFv3 neighbor relationship reestablishment or setting the link cost to the maximum value.

#### Usage Scenario

If an interface carrying OSPFv3 services alternates between up and down, OSPFv3 neighbor relationship flapping occurs on the interface. During the flapping, OSPFv3 frequently sends Hello packets to reestablish the neighbor relationship, synchronizes LSDBs, and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, OSPFv3 services, and other OSPFv3-dependent services. OSPFv3 neighbor relationship flapping suppression can address this problem by delaying OSPFv3 neighbor relationship reestablishment or preventing service traffic from passing through flapping links.

#### Pre-configuration Tasks

Before configuring OSPFv3 neighbor relationship flapping suppression, complete the following tasks:

- Configure an IP address for each interface to ensure that neighboring devices are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

#### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** (Optional) Disable OSPFv3 neighbor relationship flapping suppression globally.

- Run the **ospfv3 [ process-id ]** command to enter the OSPFv3 view.
- Run the **suppress-flapping peer disable** command to disable OSPFv3 neighbor relationship flapping suppression.

To disable the function globally, perform this step.

**Step 3** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 4** (Optional) Run **ospfv3 suppress-flapping peer disable**

OSPFv3 neighbor relationship flapping suppression is disabled on the interface.

To disable the function on one of the interfaces, perform this step.

**Step 5** Configure an OSPFv3 neighbor relationship flapping suppression mode, which can be Hold-down or Hold-max-cost mode.

- Hold-down mode:

Run the **ospfv3 suppress-flapping peer hold-down interval [ instance instance-id ]** command to configure the Hold-down mode and set its duration.

If flooding and topology changes frequently occur during OSPFv3 neighbor relationship establishment, you can configure the Hold-down mode to prevent OSPFv3 neighbor relationship reestablishment within a period of time. This prevents frequent LSDB synchronization and a large number of packets from being exchanged.

- Hold-max-cost mode:

If user service traffic is frequently switched, you can configure OSPFv3 neighbor relationship flapping suppression to work in Hold-max-cost mode. In this mode, the link cost is set to the maximum value (Max-cost) within a period of time, preventing user service traffic from passing through flapping links.

- a. (Optional) Set the maximum cost for OSPFv3 links. If the value needs to be modified, perform this step.

- i. Run **quit**

Return to the system view.

- ii. Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

- iii. Run **maximum-link-cost cost**

The maximum cost of OSPFv3 is set.

- iv. Run **quit**

Return to the system view.

- v. Run **interface interface-type interface-number**

The interface view is displayed.

- b. (Optional) Run **ospfv3 suppress-flapping peer hold-max-cost disable [ instance instance-id ]**

The Hold-max-cost neighbor relationship flapping mode is disabled.

To disable the Hold-max-cost neighbor relationship flapping suppression mode, perform this step.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

**Step 6** (Optional) Run **ospfv3 suppress-flapping peer { detecting-interval *detecting-interval* | threshold *threshold* | resume-interval *resume-interval* }\* [ instance *instance-id* ]**

Detection parameters are configured for OSPFv3 neighbor relationship flapping suppression.

Each OSPFv3 interface on which OSPFv3 neighbor relationship flapping suppression is enabled starts a flapping counter. If the interval between two successive neighbor status changes from Full to a non-Full state is shorter than *detecting-interval*, a valid flapping\_event is recorded, and the flapping\_count is incremented by 1.

When the flapping\_count reaches or exceeds *threshold*, flapping suppression takes effect.

If the interval between two successive neighbor status changes from Full to a non-Full state is longer than *resume-interval*, the flapping\_count is reset.

 **NOTE**

The value of *resume-interval* must be greater than that of *detecting-interval*.

You can configure detection parameters for OSPFv3 neighbor relationship flapping suppression on specific interfaces according to network conditions. However, using the default values of these parameters is recommended.

**Step 7** Run **quit**

Return to the system view.

**Step 8** Run **commit**

The configuration is committed.

**Step 9** Run **quit**

Return to the user view.

**Step 10** (Optional) Run **reset ospfv3 *process-id* suppress-flapping peer [ interface-type *interface-number* ] [ notify-peer ]**

The OSPFv3 interface is configured to exit neighbor relationship flapping suppression.

To exit neighbor relationship flapping suppression in advance after the fault that causes OSPFv3 neighbor relationship flapping is rectified, perform this step.

 NOTE

Interfaces exit flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding OSPFv3 process is reset.
- An OSPFv3 neighbor relationship is reset using the **reset ospfv3 peer** command.
- OSPFv3 neighbor relationship flapping suppression is disabled globally using the **suppress-flapping peer disable** command in the OSPFv3 view.
- The **reset ospfv3 suppress-flapping peer** command is run to exit flapping suppression.

----End

## Verifying the Configuration

Run the **display ospfv3 [ process-id ] interface [ no-peer | area area-id ] [ interface-type interface-number ]** command to check the status of OSPFv3 neighbor relationship flapping suppression.

### 1.1.5.2.10 Configuring Routing Loop Detection for Routes Imported to OSPFv3

Routing loop detection for routes imported into OSPFv3 enables a device to check whether routes advertised by the device are imported back. If routes advertised by the device are imported back, the device advertises a large link cost for the routes to prevent routing loops.

## Usage Scenario

Routing loops may occur when an OSPFv3 process imports routes. If routing loop detection is enabled for routes imported to OSPFv3 on a device and this device detects that it imports a route advertised by itself, it sends this route with a large link cost to other devices. After receiving this route, these devices preferentially select other paths, thereby preventing routing loops.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 (Optional) Run **clear route loop-detect ospfv3 alarm-state**

The routing loop detection alarm state exits, and related alarms are cleared.

 NOTE

After detecting an OSPFv3 routing loop, the device reports an alarm. Because the device cannot automatically detect whether the routing loop is eliminated, you need to run this command after the routing loop is eliminated to prevent the device from advertising a large cost for imported routes and manually clear the OSPFv3 routing loop alarm. If this command is executed when the routing loop has not been eliminated, the alarm is reported again.

### Step 3 Run **route loop-detect ospfv3 enable**

Routing loop detection is enabled for routes imported to OSPFv3.

 NOTE

To reduce the impact of routing loops on services, you are advised to enable routing loop detection for routes imported to OSPFv3.

**Step 4 Run commit**

The configuration is committed.

----End

### 1.1.5.2.11 Disabling OSPFv3 Interface Flapping Suppression

OSPFv3 interface flapping suppression is globally enabled by default. If this function is not needed, you can disable it.

#### Usage Scenario

If OSPFv3 interfaces frequently alternate between up and down, the interfaces will flap, and protocol packets will be frequently exchanged, affecting OSPFv3 services and other services relying on OSPFv3. Interface flapping suppression can address this issue. This function allows a device to delay interface state changes to up. If this function is not needed, you can disable it.

#### Pre-configuration Tasks

Before configuring OSPFv3 interface flapping suppression, [configure basic OSPFv3 functions](#).

#### Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run ospfv3 suppress-flapping interface disable**

OSPFv3 interface flapping suppression is disabled.

**Step 3 Run commit**

The configuration is committed.

----End

#### Checking the Configuration

Run the **display current-configuration configuration ospfv3** command to check the status of OSPFv3 interface flapping suppression.

### 1.1.5.2.12 Configuring OSPFv3 Flush Source Tracing

OSPFv3 flush source tracing helps improve fault locating efficiency.

#### Usage Scenario

If network-wide OSPFv3 LSAs are deleted, flush LSAs are flooded, which adversely affects network stability. In this case, source tracing must be implemented to

locate the root cause of the fault immediately to minimize the impact. However, OSPFv3 itself does not support source tracing. A conventional solution is isolation node by node until the faulty node is located, but the solution is complex and time-consuming. To address this problem, enable OSPFv3 flush source tracing.

 NOTE

The following steps are optional, and choose them as required.

## Pre-configuration Tasks

Before configuring OSPFv3 flush source tracing, complete the following task:

- Configure an IP address for each interface to ensure that neighboring devices are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** (Optional) Run **ospfv3 flush-source-trace disable**

OSPFv3 flush source tracing is disabled globally.

To disable OSPFv3 flush source tracing globally, perform this step.

**Step 3** Run **interface *interface-type* *interface-number***

The interface view is displayed.

**Step 4** (Optional) Run **ospfv3 flush-source-trace block**

OSPFv3 flush source tracing is disabled on the interface.

To disable OSPFv3 flush source tracing on an interface, perform this step.

**Step 5** Run **quit**

Return to the system view.

**Step 6** Run **ospfv3 [ *process-id* ] [ **vpn-instance** *vpn-instance-name* ]**

The OSPFv3 process is started, and the OSPFv3 view is displayed.

**Step 7** Run **quit**

Return to the system view.

**Step 8** Run **ospfv3 flush-source-trace port *port-number***

A UDP port number is configured for OSPFv3 flush source tracing packets.

An OSPFv3 flush source tracing port is used to send and receive OSPFv3 flush source tracing packets and is represented by a UDP port number. If the port number used by UDP packets conflicts with that of another application, perform this step.

### Step 9 Run commit

The configuration is committed.

### Step 10 Run quit

Return to the user view.

### Step 11 Run **reset ospfv3 [ process-id ] flush-source-trace**

OSPFv3 flush source tracing is reset.

If a large number of OSPFv3 flush source tracing statistics exist on a device, you can run the **reset ospfv3 flush-source-trace** command to reset the statistics and restart source tracing. If the command is run, all dynamic source tracing statistics are deleted, and the device needs to re-negotiate the source tracing capability with neighboring devices.

----End

## Verifying the Configuration

Run the **display ospfv3 [ process-id ] [ area area-id-integer ] flush-source-trace analysis-info** command to check information about determined and suspected faulty nodes identified by OSPFv3 flush source tracing.

### 1.1.5.2.13 Configuring an OSPFv3 Hostname

Compared with router IDs, OSPFv3 hostnames are easier to memorize. Therefore, using hostnames to identify routers can facilitate network management.

## Usage Scenario

To facilitate network management, configure dynamic OSPFv3 hostnames to identify routers.

Either dynamic or static OSPFv3 hostnames can be configured. In dynamic mode, a hostname can be configured on and advertised by the local device. The mapping between the local device's router ID and hostname can then be queried on a remote router that successfully learns this dynamic hostname.

## Pre-configuration Tasks

Before configuring a hostname, complete the following tasks:

- Configure an IP address for each interface to ensure that neighboring routers can use the IP addresses to communicate with each other.
- Configure basic OSPFv3 functions.**

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **hostname host-name**

A dynamic OSPFv3 hostname is configured.

#### NOTE

If the *host-name* parameter is specified, the value of *host-name* is advertised as the dynamic hostname. If only the **hostname** command is run and *host-name* is not specified, the device name specified in the **sysname** command is advertised as the dynamic hostname.

### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the following command to check dynamic OSPFv3 hostnames:

- **display ospfv3 [ process-id ] hostname-table**

### 1.1.5.2.14 Configuring an OSPFv3 Stub Area

A non-backbone area in the edge of an AS can be configured as a stub area so that routes from other areas on the OSPFv3 network or AS external routes are not transmitted. This reduces bandwidth and storage resource consumption on the router, routing table size, and the amount of routing information to be transmitted.

## Usage Scenario

The number of LSAs can be reduced by partitioning an AS into different areas. To reduce the number of entries in the routing table and the number of LSAs to be transmitted in a non-backbone area, configure the non-backbone area on the border of the AS as a stub area.

Configuring a stub area is optional. A stub area generally resides on the border of an AS. For example, a non-backbone area with only one ABR can be configured as a stub area. In a stub area, the number of entries in the routing table and the amount of routing information to be transmitted greatly decrease.

Note the following points when configuring a stub area:

- The backbone area (Area 0) cannot be configured as a stub area.
- If an area needs to be configured as a stub area, all the routers in this area must be configured with stub attributes using the **stub** command.
- An ASBR cannot exist in a stub area. External routes are not transmitted in the stub area.

## Pre-configuration Tasks

Before configuring a stub area, complete the following tasks:

- Configuring IP addresses for interfaces to ensure that neighboring routers are reachable at the network layer

- **Configuring Basic OSPFv3 Functions**

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

**Step 3** Run **area area-id**

The OSPFv3 area view is displayed.

**Step 4** Run **stub**

The specified area is configured as a stub area.

 **NOTE**

- All routers in a stub area must be configured with stub attributes using the **stub** command.
- Configuring or deleting stub attributes will update routing information in the area. Stub attributes can be deleted or configured again only after the routing update is complete.

**Step 5** (Optional) Run **stub no-summary**

The ABR is prevented from sending Type 3 LSAs (Summary LSAs) to the stub area.

**Step 6** (Optional) Run **stub default-route-advertise backbone-peer-ignore**

The device is configured to advertise default routes and ignore the status of the neighbors in the backbone area.

**Step 7** (Optional) Run **default-cost cost**

The cost of the default route to the stub area is set.

To ensure the reachability of AS external routes, the ABR in the stub area generates a default route and advertises the route to the other routers in the stub area.

**Step 8** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the **display ospfv3 [ process-id ] lsdb** command, and you can view LSDB information, including the types of LSAs and link state IDs.

When Router is in a common area, there are AS external routes in the routing table. After the area where Router resides is configured as a stub area, AS external routes are invisible.

### 1.1.5.2.15 Configuring an OSPFv3 NSSA

By configuring a non-backbone area on the border of an autonomous system (AS) as a not-so-stubby area (NSSA), you can reduce the size of the routing table and the amount of routing information to be transmitted.

#### Usage Scenario

An excessive number of entries in a routing table wastes network resources and causes high CPU usage. To reduce the number of entries in a routing table and the amount of routing information to be transmitted, configure a non-backbone area on the border of an AS as a stub area or an NSSA. For details about how to configure an OSPFv3 stub area, see [1.1.5.2.14 Configuring an OSPFv3 Stub Area](#).

OSPFv3 stub areas cannot import external routes from outside an AS, nor can they learn the external routes imported by the other areas in the same AS. To import external routes to an area and minimize resource consumption, configure the area as an NSSA. NSSAs can import the external routes (outside an AS) and advertise them within the entire AS, without learning external routes from the other areas in the AS. This reduces the consumption of bandwidth and storage resources on the router.

NSSA attributes must be configured on all routers in an NSSA.

#### Pre-configuration Tasks

Before configuring an OSPFv3 NSSA, complete the following tasks:

- Configure an IP address for each interface to ensure that neighboring devices can communicate with each other at the network layer.
- [Configure basic OSPFv3 functions](#).

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

##### Step 3 (Optional) Run **lsa-forwarding-address { standard | zero-translate }**

The OSPFv3 forwarding address (FA) function is enabled.

##### Step 4 Run **area area-id**

The OSPFv3 area view is displayed.

##### Step 5 Run **nssa [ { default-route-advertise [ backbone-peer-ignore | cost cost-value | type type-value | tag tag-value ] \* } | no-import-route | no-summary | translator-always | translator-interval interval-value | set-n-bit | suppress-forwarding-address ] \***

The specified area is configured as an NSSA.

The usage scenarios of the **nssa** command parameters are as follows:

- To allow default Type 7 LSA generation, specify **default-route-advertise**. After this parameter is specified, a Type 7 LSA default route will be generated on an ABR, regardless of whether the route ::/0 exists in the ABR's routing table. On an ASBR, however, a Type 7 LSA default route can be generated only if the route ::/0 exists in the ASBR's routing table.
- If an ASBR also functions as an ABR, specifying **no-import-route** prevents OSPFv3 from advertising the external routes imported using the **import-route** command to the NSSA.
- To reduce the number of LSAs to be transmitted to the NSSA, specify **no-summary** on an ABR. This prevents the ABR from transmitting Summary LSAs (Type 3 LSAs) to the NSSA.

 **NOTE**

After the **nssa default-route-advertise backbone-peer-ignore no-summary** command is run, the ABR generates both the default Type 7 LSA and default Type 3 LSA as long as the backbone area contains interfaces that are up, regardless of whether neighbor relationships in Full state exist. In this case, the default Type 3 LSA preferentially takes effect.

- After **set-n-bit** is set, the DD packets sent by the router carry the N-bit of 1.
- If multiple ABRs are deployed in the NSSA, OSPFv3 automatically selects an ABR (generally the ABR with the largest router ID) as a translator to translate Type 7 LSAs into Type 5 LSAs. Each change of a translator causes LSA flooding. To prevent this issue, specify **translator-always** on an ABR to configure the ABR as a fixed translator. To allow two ABRs to participate in load balancing, configure **translator-always** on the ABRs.
- To ensure service continuity during translator switching, specify the **translator-interval** parameter. The value of *interval-value* must be greater than the flooding interval.
- To disable translated Type 5 LSAs from carrying an FA, specify the **suppress-forwarding-address** parameter.

**Step 6 Run commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display ospfv3 [ process-id ] routing [ [ ipv6-address prefix-length ] | abr-routes | asbr-routes | ase-routes | inter-routes | intra-routes | nssa-routes ] [ verbose ]** command to check the OSPFv3 routing table information.
- Run the **display ospfv3 [ process-id ] lsdb [ area area-id ] [ [ originate-router advertising-router-id | hostname hostname ] | self-originate ] { grace | inter-prefix | inter-router | intra-prefix | link | network | router | router-information | nssa } [ link-state-id ] [ age { min-value min-age-value | max-value max-age-value } \* ]** command to check the OSPFv3 LSDB information.

### 1.1.5.2.16 Configuring OSPFv3 IP FRR

If a link fails, OSPFv3 IP FRR rapidly switches traffic to a backup link, which prevents traffic interruption and improves OSPFv3 network reliability.

#### Usage Scenario

As networks develop, services such as Voice over IP (VoIP) and on-line video services require high-quality real-time transmission. However, OSPFv3 takes more than 50 ms to rectify a fault, which cannot meet the requirement for real-time transmission of these services.

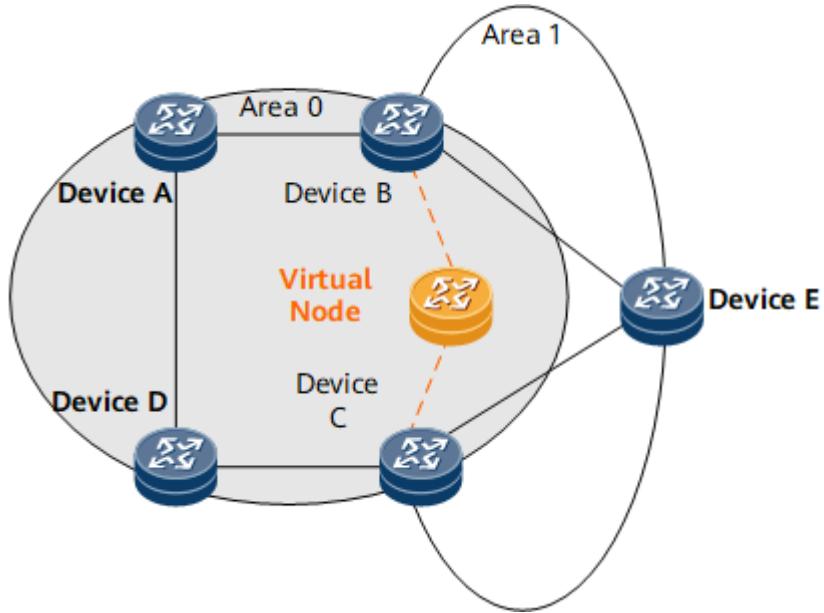
Nevertheless, if an OSPFv3 fault occurs, traffic can be switched to a new link only after the fault detection that lasts several milliseconds, fault notification to the routing control plane that lasts several milliseconds, new topology information generation and flooding that last tens of milliseconds, Shortest Path First (SPF) calculation that lasts tens of milliseconds, and new route notification and adding that last hundreds of milliseconds.

OSPFv3 IP FRR calculates a backup link in advance. If the primary link fails, OSPFv3 IP FRR rapidly switches traffic to the backup link without interrupting traffic. This protects traffic and greatly improves OSPFv3 network reliability.

OSPFv3 IP FRR is applicable to services that are sensitive to the packet delay and packet loss.

OSPFv3 LFA FRR uses the SPF algorithm to calculate the shortest path from each neighbor (root node) that provides a backup link to the destination node and store the node-based backup next hop, which applies to single-node routing scenarios. As networks are increasingly diversified, two ABRs or ASBRs are deployed to improve network reliability. In this case, OSPFv3 FRR in a scenario where multiple nodes advertise the same route is needed. In [Figure 1-186](#), Device B and Device C function as ABRs to forward area 0 and area 1 routes. Device E advertises an intra-area route. Upon receipt of the route, Device B and Device C translate it into a Type 3 LSA and flood the LSA to area 0. After OSPFv3 FRR is enabled on Device A, Device A considers Device B and Device C as its neighbors. Without a fixed neighbor as the root node, Device A fails to calculate FRR backup next hop. To address this problem, a virtual node is simulated between Device B and Device C and used as the root node of Device A, and Device A uses the LFA algorithm to calculate the backup next hop. This solution converts multi-node routing into single-node routing.

**Figure 1-186 OSPFv3 FRR in the scenario where multiple nodes advertise the same route**



After OSPFv3 IP FRR is configured, the lower layer needs to fast respond to a link change so that traffic can be fast switched to the backup link. After FRR and BFD are bound, link failures can be detected rapidly so that traffic is rapidly switched to the backup link if the primary link fails.

## Pre-configuration Tasks

Before configuring OSPFv3 IP FRR, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

## Enabling OSPFv3 IP FRR

With OSPF IP FRR and loop-free backup links, a device can switch traffic to a backup link immediately if the primary link fails.

## Context

Perform the following steps on the router:

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ] [ vpn-instance vpnname ]**

An OSPFv3 process is enabled, and the OSPFv3 view is displayed.

### Step 3 Run frr

The OSPFv3 IP FRR view is displayed.

### Step 4 Run loop-free-alternate

OSPFv3 IP FRR is enabled, and a loop-free backup link is generated.



OSPFv3 can generate a loop-free backup link only when the OSPFv3 IP FRR traffic protection inequality is met.

### Step 5 (Optional) Run **frr-policy route { route-policy route-policy-name | route-filter route-filter-name }**

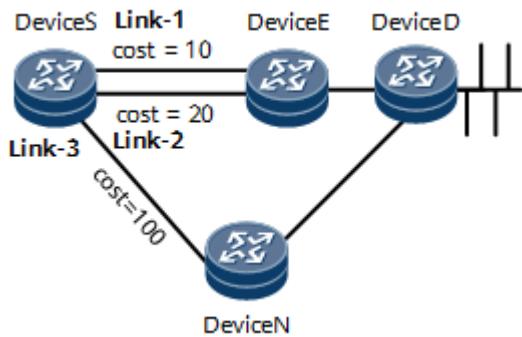
An OSPFv3 IP FRR filtering policy is configured.

After the OSPFv3 IP FRR filtering policy is configured, only the OSPFv3 backup routes that match the filtering rules in the policy can be added to the forwarding table.

### Step 6 (Optional) Run **tiebreaker { node-protecting | lowest-cost } preference preference**

The solution of selecting a backup path for OSPFv3 IP FRR is configured. By default, the solution of selecting a backup path for OSPFv3 IP FRR is node-protection path first. In actual networking scenarios, the solution may need to be changed to smallest-cost path first due to considerations such as interface forwarding capacity and link cost. In **Figure 1-187**, the primary path is Link-1 (DeviceS -> DeviceE -> DeviceD), and Link-2 and Link-3 (DeviceS -> DeviceN -> DeviceD) are backup path candidates. By default, Link-3 is selected as the backup path. To change the solution of selecting a backup path to smallest-cost path first, run the **tiebreaker** command. After the command is run, Link-2 is selected as the backup path.

**Figure 1-187** Solution of selecting a backup path for OSPFv3 IP FRR



### Step 7 Run commit

The configuration is committed.

----End

## (Optional) Binding IP FRR and BFD

Binding IP FRR and BFD enables the lower layer to fast respond to a link change so that traffic can be rapidly switched to the backup link if the primary link fails.

### Context

Binding the BFD session status to the link status of an interface ensures that link failures are detected rapidly and that traffic is rapidly switched to the backup link.

- If IP FRR of an OSPFv3 process is bound to BFD, IP FRR on all interfaces in the OSPFv3 process is bound to BFD.
- If only a small number of interfaces need to have IP FRR bound to BFD, bind IP FRR on each interface to BFD one by one.

Perform the following steps on the router where IP FRR and BFD need to be associated:

### Procedure

- Bind IP FRR and BFD in an OSPFv3 process.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **ospfv3**  
An OSPFv3 process is enabled, and the OSPFv3 view is displayed.
  - c. Run **bfd all-interfaces frr-binding**  
IP FRR and BFD are bound in the OSPFv3 process.
  - d. Run **commit**  
The configuration is committed.
- Bind IP FRR and BFD on a specified OSPFv3 interface.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface *interface-type interface-number***  
An interface view is displayed.
  - c. Run **ospfv3 bfd frr-binding**  
IP FRR and BFD are bound on the interface.
  - d. Run **commit**  
The configuration is committed.

----End

### Follow-up Procedure

The BFD configuration on an interface takes precedence over that in an OSPFv3 process. If BFD is enabled on an interface, a BFD session is established based on the BFD parameters set on the interface.

## (Optional) Blocking FRR on an OSPFv3 Interface

If FRR is not required on certain OSPFv3 interfaces, FRR needs to be blocked on these interfaces.

### Context

If an interface runs key services, ensure that the backup path does not pass through this interface so that services will not be affected after FRR calculation.

Perform the following steps on the interfaces of the device where OSPFv3 IP FRR has been configured:

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface *interface-type* *interface-number***

The view of the OSPFv3 interface running FRR is displayed.

#### Step 3 Run **ospfv3 frr block**

FRR is blocked on the OSPFv3 interface.

#### Step 4 Run **commit**

The configuration is committed.

----End

### Verifying the Configuration of OSPFv3 IP FRR

After configuring OSPFv3 IP FRR, you can view information about the primary and backup links.

### Prerequisites

OSPFv3 IP FRR has been configured.

### Procedure

- Run the **display ospfv3 [ *process-id* ] routing verbose** command to view the primary and backup links after OSPFv3 IP FRR is enabled.

----End

### 1.1.5.2.17 Configuring BFD for OSPFv3

To speed up OSPFv3 convergence when the link status changes, you can configure BFD for OSPFv3. After detecting a link failure, BFD notifies the routing protocol of the failure, which triggers fast convergence. When the neighbor relationship is Down, the BFD session is deleted dynamically.

## Usage Scenario

To speed up OSPFv3 convergence when the link status changes, you can configure BFD for OSPFv3 links.

BFD detects links much faster than keep-alive protocols do. If OSPFv3 is bound to BFD sessions, BFD can notify OSPFv3 of link failures immediately, and then OSPFv3 perform route calculation and convergence in the new network topology.

## Pre-configuration Tasks

Before configuring BFD for OSPFv3, complete the following tasks:

- Configure a link layer protocol.
- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

## Configuring BFD Globally

On the two devices that need to establish a BFD session, you can configure BFD for all the interfaces in a certain OSPFv3 process.

### Context

Perform the following steps on the router that runs OSPFv3:

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bfd**

BFD is enabled globally.

#### Step 3 Run **commit**

The configuration is committed.

----End

## Configuring BFD for OSPFv3

After enabling BFD for OSPFv3, you need to configure BFD parameters in the OSPFv3 process.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **bfd all-interfaces enable**

BFD is enabled for the OSPFv3 process to create BFD sessions.

### Step 4 (Optional) Run **bfd all-interfaces { min-transmit-interval *Tx-Value* | min-receive-interval *Rx-Value* | detect-multiplier *Mul-Value* }<sup>\*</sup>**

BFD parameters are configured for the OSPFv3 process.

The default interval at which BFD packets are transmitted and the default detection multiplier are recommended. As such, this step can be skipped.

The parameters need to be configured based on network conditions and requirements on network reliability. A short transmission interval for BFD packets can be set for a link that requires higher reliability, and a long transmission interval can be used for a link that has low reliability requirements.

#### NOTE

- Actual interval at which BFD packets are transmitted on the local device = Max { *Tx-Value* (interval at which BFD packets are transmitted) set on the local device, *Rx-Value* (interval at which BFD packets are received) set on the peer device }
- Actual interval at which BFD packets are received on the local device = Max { *Tx-Value* (interval at which BFD packets are transmitted) set on the peer device, *Rx-Value* (interval at which BFD packets are received) set on the local device }
- Actual period for BFD detection on the local device = Actual interval at which BFD packets are received on the local device x Detection multiplier specified by *Mul-Value* on the peer device

For example:

- On the local device, the interval at which packets are transmitted is set to 200 ms, the interval at which packets are received is set to 300 ms, and the detection multiplier is set to 4.
- On the peer device, the interval at which packets are transmitted is set to 100 ms, the interval at which packets are received is set to 600 ms, and the detection multiplier is set to 5.

Then:

- On the local device, the actual interval at which BFD packets are transmitted is 600 ms (calculated by Max { 200 ms, 600 ms }); the actual interval at which BFD packets are received is 300 ms (calculated by Max { 100 ms, 300 ms }); the actual detection period is 1500 ms (calculated by 300 ms x 5).
- On the peer device, the actual interval at which BFD packets are transmitted is 300 ms (calculated by Max { 100 ms, 300 ms }); the actual interval at which BFD packets are received is 600 ms (calculated by Max { 200 ms, 600 ms }); the actual detection period is 2400 ms (calculated by 600 ms x 4).

### Step 5 (Optional) Disable a specified interface from creating a BFD session.

After BFD is configured for an OSPFv3 process, all interfaces on which neighbor relationships are in Full state in the OSPFv3 process create BFD sessions. If BFD is not required on specific interfaces, disable these interfaces from creating BFD sessions.

1. Run **quit**

Return to the system view.

2. Run **interface *interface-type* *interface-number***

The interface view is displayed.

3. Run **ospfv3 bfd block**

The interface is disabled from creating a BFD session.

4. Run **quit**

Return to the system view.

5. Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

**Step 6** (Optional) Run **bfd all-interfaces incr-cost { cost | max-reachable } [ wtr <wtr-value> ]**

The OSPFv3 process is enabled to adjust the cost based on the status of the associated BFD session.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

The delay configured for an interface to restore the cost based on the BFD session status takes precedence over that configured for a process to restore the cost based on the BFD session status.

**Step 7** Run **commit**

The configuration is committed.

----End

## (Optional) Configuring BFD for a Specified Interface

To configure BFD only on a specified interface, or enable an interface to detect link failures faster after BFD for OSPFv3 is enabled globally, configure BFD on the specified interface.

## Context

To configure BFD only on a specified interface, or enable an interface to detect link failures faster after BFD for OSPFv3 is enabled globally, perform the following steps on the interface:

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface *interface-type* *interface-number***

The interface view is displayed.

**Step 3** Run **ospfv3 bfd enable [ instance *instance-id* ]**

BFD is enabled on the interface to create BFD sessions.

When BFD is configured globally and the neighbor state is Full, BFD sessions are established using default BFD parameters.

To set parameters for BFD sessions as required, run the **ospfv3 bfd { min-transmit-interval *min-transmit-value* | min-receive-interval *min-receive-value* | detect-multiplier *detect-multiplier-value* | frr-binding } \* [ instance *instance-id* ] }** command.

If BFD for OSPFv3 is configured for an Eth-Trunk with multiple physical interfaces added in a VLAN and **per-link one-arm-echo** is not specified, the BFD session may go down even if only one of the physical interfaces goes down. As a result, the OSPFv3 neighbor relationship also goes down. If **per-link one-arm-echo** is specified in this case, the BFD session goes down only if all the physical interfaces are down, which ensures that the OSPFv3 neighbor relationship can be established normally.

#### NOTE

- The configuration of BFD on an interface takes precedence over that in a process. That is, if BFD session parameters are configured for both a process and an interface, the parameters on the interface will be used to establish BFD sessions on the interface.
- If the parameters of BFD sessions are set but the **ospfv3 bfd enable** command is not run, BFD is not enabled on the interface.

The default interval at which BFD packets are transmitted and the default detection multiplier are recommended. As such, this step can be skipped.

The parameters need to be configured based on network conditions and requirements on network reliability. A short transmission interval for BFD packets can be set for a link that requires higher reliability, and a long transmission interval can be used for a link that has low reliability requirements.

#### NOTE

- Actual interval at which BFD packets are transmitted on the local device = Max { *min-transmit-value* (interval at which BFD packets are transmitted) set on the local device, *min-receive-value* (interval at which BFD packets are received) set on the peer device }
- Actual interval at which BFD packets are received on the local device = Max { *min-transmit-value* (interval at which BFD packets are transmitted) set on the peer device, *min-receive-value* (interval at which BFD packets are received) set on the local device }
- Actual period for BFD detection on the local device = Actual interval at which BFD packets are received on the local device x Detection multiplier specified by *detect-multiplier-value* on the peer device

For example:

- On the local device, the interval at which BFD packets are transmitted is set to 200 ms, the interval at which BFD packets are received is set to 300 ms, and the detection multiplier is set to 4.
- On the peer device, the interval at which BFD packets are transmitted is set to 100 ms, the interval at which BFD packets are received is set to 600 ms, and the detection multiplier is set to 5.

Then:

- On the local device, the actual interval at which BFD packets are transmitted is 600 ms (calculated through Max { 200 ms, 600 ms }); the actual interval at which BFD packets are received is 300 ms (calculated through Max { 100 ms, 300 ms }); the actual detection period is 1500 ms (calculated through 300 ms x 5).
- On the peer device, the actual interval at which BFD packets are transmitted is 300 ms (calculated through Max { 100 ms, 300 ms }); the actual interval at which BFD packets are received is 600 ms (calculated through Max { 200 ms, 600 ms }); the actual detection period is 2400 ms (calculated through 600 ms x 4).

**Step 4** (Optional) Run **ospfv3 bfd incr-cost { cost | max-reachable } [ wtr <wtr-value> ]**

The OSPFv3 interface is enabled to adjust the cost based on the BFD session status.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

**Step 5** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of BFD for OPSFv3

After configuring BFD for OSPFv3, verify information about the BFD session.

## Prerequisites

BFD for OSPFv3 has been configured.

## Procedure

- Run the **display ospfv3 [ process-id ] bfd session [ interface-type interface-number ] [ neighbor-id ] [ verbose ]** command to check information about the BFD session.

----End

### 1.1.5.2.18 Configuring OSPFv3 Authentication

Configuring OSPFv3 authentication enables a device to authenticate sent and received OSPFv3 packets, protecting the device against attacks through forged OSPFv3 packets.

## Usage Scenario

OSPFv3 IPsec uses a complete set of IPsec mechanisms to authenticate sent and received OSPFv3 packets, protecting devices against forged OSPFv3 packets.

However, on some special networks, a mobile ad hoc network (MANET) for example, IPsec is difficult to deploy and maintain. To address this problem, Authentication Trailer for OSPFv3 can be used to implement authentication.

If Authentication Trailer for OSPFv3 is used, an authentication field is added to each OSPFv3 packet for encryption. When receiving an OSPFv3 packet from a remote device, the local device discards the packet if the authentication password carried in the packet is different from the local one, protecting the local device against potential attacks.

## Pre-configuration Tasks

Before configuring OSPFv3 authentication, complete the following tasks:

- Enable IPv6.
- **Configure basic OSPFv3 functions.**
- If keychain authentication needs to be used, configure basic keychain functions.

## Configuring OSPFv3 IPsec

OSPFv3 IPsec provides a set of IPsec mechanisms to authenticate sent and received OSPFv3 packets, protecting devices against invalid OSPFv3 packets.

## Usage Scenario

OSPFv3 IPsec uses a set of IPsec mechanisms to authenticate sent and received OSPFv3 packets, protecting devices against invalid OSPFv3 packets.

### ?1. Configuring an IPsec Proposal

Proposal defines encryption and authentication algorithms to authenticate OSPFv3 protocol packets.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run ipsec proposal *proposal-name*

A security proposal is created and the security proposal view is displayed.

### Step 3 Run encapsulation-mode transport

The protocol packet encapsulation mode is configured.

### Step 4 (Optional) Run transform { ah | esp | ah-esp }

A security protocol is configured.



Because AH does not support encryption in IPsec scenarios, ESP is recommended.

### Step 5 An authentication algorithm and an encryption algorithm are configured based on the selected security protocol.

- If Authentication Header (AH) is configured, run the **ah authentication-algorithm { md5 | sha1 | sha2-256 | sha2-384 | sha2-512 }** command to configure an authentication algorithm.



To help provide high security, do not use the MD5 or SHA1 algorithm as an AH authentication algorithm.

- If ESP is configured, run the **esp authentication-algorithm { md5 | sha1 | sha2-256 | sha2-384 | sha2-512 }** command to configure an authentication algorithm.

 **NOTE**

To help provide high security, do not use the MD5 or SHA1 algorithm as an ESP authentication algorithm.

- If ESP is configured, run the **esp encryption-algorithm { des | 3des | aes { 128 | 192 | 256 } }** command to configure an ESP encryption algorithm.

 **NOTE**

To help provide high security, do not use the DES or 3DES algorithm as an ESP encryption algorithm.

**Step 6 Run commit**

The configuration is committed.

----End

## ??. Configuring an IPsec SA

To create a manual IPsec tunnel, you need to use the SPI, string-key, authentication-hex, or encryption-hex.

### Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run ipsec sa *sa-name***

An SA is created, and the SA view is displayed.

**Step 3 Run proposal *proposal-name***

A security proposal is applied to the SA.

 **NOTE**

A security proposal must be configured before it can be associated with protocol packet flows.

One SA can use only one security proposal. If a security proposal has been applied to an SA, the SA can use another security proposal only after the original one is deleted.

**Step 4 Run sa spi { inbound | outbound } { ah | esp } *spi-number***

An SPI is set.

 **NOTE**

The SPI uniquely identifies an SA. The inbound and outbound SPIs are set, and the inbound SPI on the local end must be the same as the outbound SPI on the peer end.

**Step 5 To configure the authentication key, run either of the following commands:**

1. Run **sa authentication-hex { inbound | outbound } { ah | esp } [ cipher ] *key-cipher-key***

- An authentication key in hexadecimal format or ciphertext is set.
2. Run **sa string-key { inbound | outbound } { ah | esp } [ cipher ] string-cipher-key**

An authentication key in string format is set.

 NOTE

An authentication key for outgoing protocol packets on the local end must be identical with that for incoming protocol packets on the peer end.

If multiple authentication keys are configured, the latest one takes effect.

Updating keys periodically is recommended.

- Step 6** (Optional) Run **sa encryption-hex { inbound | outbound } esp [ cipher ] hex-cipher-key**

An encryption key is set.

- Step 7** Run **commit**

The configuration is committed.

----End

### ?.3. Enabling OSPFv3 IPsec

A Security Association (SA) configured in an OSPFv3 process or OSPFv3 area is used to authenticate packets of the process.

#### Context

Perform the following steps on the router that runs OSPFv3:

#### Procedure

- Enable IPsec in an OSPFv3 process.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.
  - c. Run **ipsec sa sa-name**

An SA is configured in the process.

An OSPFv3 process can be associated with multiple OSPFv3 areas. An SA applied in the OSPFv3 process can be used in the associated areas.
  - d. Run **commit**

The configuration is committed.
- Enable IPsec in an OSPFv3 area.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

- c. Run **area area-id**

The OSPFv3 area view is displayed.

- d. Run **ipsec sa sa-name**

An SA is configured in the OSPFv3 area.

 NOTE

The SA configured on an OSPFv3 area takes precedence over that configured in an OSPFv3 process.

- e. Run **commit**

The configuration is committed.

- Enable IPsec in an OSPFv3 area.

- a. Run **system-view**

The system view is displayed.

- b. Run **interface interface-type interface-number**

The interface view is displayed.

- c. Run **ospfv3 ipsec sa sa-name**

An SA in the view of an OSPFv3 interface is configured.

 NOTE

- The SA configured in the interface view takes precedence over that configured in the OSPFv3 area view or the OSPFv3 process view.
- The **ospfv3 ipsec sa** command can be used on all the OSPFv3 instances of an interface.

- d. Run **commit**

The configuration is committed.

----End

## Configuring OSPFv3 Authentication Trailer

Open Shortest Path First version 3 (OSPFv3) supports packet authentication, enabling OSPFv3 devices to receive only the OSPFv3 packets that are authenticated. If packets fail to be authenticated, OSPFv3 neighbor relationships cannot be established. This section describes how to configure an authentication mode.

## Usage Scenario

OSPFv3 authentication trailer supports HMAC-SHA256 authentication and HMAC-SM3 authentication.

 NOTE

By default, authentication is not configured for OSPFv3 areas, processes, or interfaces. You are advised to configure an authentication mode to ensure system security.

For security purposes, you are advised not to use weak security algorithms in OSPFv3. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

When configuring an authentication password, select the ciphertext mode because the password is saved in the configuration file as a plaintext if you select the plaintext mode, which has a high risk. To ensure device security, change the password periodically.

## Procedure

- Configure area authentication.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **ospfv3 [ process-id ]**  
The OSPFv3 view is displayed.
  - c. Run **area area-id**  
The OSPFv3 area view is displayed.
  - d. Run **authentication-mode { hmac-sha256 | hmac-sm3 } key-id KeyId { plain PlainText | [ cipher ] CipherText }**  
An authentication mode is configured for the OSPFv3 area.

 NOTE

When area authentication is used, all the routers in the area must have the same area authentication mode and password.

- e. Run **commit**  
The configuration is committed.
- Configure process authentication.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **ospfv3 [ process-id ]**  
The OSPFv3 view is displayed.
  - c. Run **authentication-mode { hmac-sha256 | hmac-sm3 } key-id KeyId { plain PlainText | [ cipher ] CipherText }**  
An authentication mode is configured for the OSPFv3 process.
  - d. Run **commit**  
The configuration is committed.
- Configure interface authentication.
  - a. Run **system-view**  
The system view is displayed.

- b. Run **interface interface-type interface-number**  
The interface view is displayed.
- c. Run **ospfv3 authentication-mode { hmac-sha256 | hmac-sm3 } key-id KeyId { plain PlainText | [ cipher ] CipherText } [ instance instanceId ]**  
An authentication mode is configured for the OSPFv3 interface.

 NOTE

Interface authentication takes precedence over area authentication. For interfaces on the same subnet, the configured authentication mode and password must be identical. This requirement does not apply to the interfaces on different subnets.

- d. Run **commit**

The configuration is committed.

----End

## Configuring OSPFv3 Process Authentication

OSPFv3 supports packet authentication, with which devices accept only the OSPFv3 packets that are authenticated. If OSPFv3 packets fail to be authenticated, OSPFv3 neighbor relationships cannot be established. Configuring OSPFv3 process authentication improves OSPFv3 network security.

## Context

With the increase in attacks on TCP/IP networks and inherent defects and flawed implementation of the TCP/IP protocol suite, the attacks have increasing impacts on the networks. Attacks on network devices may even cause a network crash or lead to network unavailability. Configuring OSPFv3 process authentication improves OSPFv3 network security.

 NOTE

By default, no authentication mode is configured for an OSPFv3 process. You are advised to configure an authentication mode to ensure system security.

OSPFv3 authentication takes effect in descending order of priority as follows: interface authentication, area authentication, and process authentication.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Configure an authentication mode for the OSPFv3 process as required.

- To configure the HMAC-SHA256 or HMAC-SM3 authentication mode for the OSPFv3 process, run the **authentication-mode { hmac-sha256 | hmac-sm3 } key-id KeyId { plain PlainText | [ cipher ] CipherText } command**.

If you choose **plain**, the password will be saved as a cleartext in the configuration file, which provokes high security risks. To improve system

- security, choose ciphertext authentication and change the password periodically.
- To configure keychain authentication for the OSPFv3 process, run the **authentication-mode { keychain *Keychain-Name* }** command.

 **NOTE**

Before configuring keychain authentication, you need to run the **keychain** command to create a keychain, and then run the **key-id**, **key-string**, and **algorithm** commands to configure a key ID, a password, and an authentication algorithm, respectively, for this keychain. Otherwise, OSPFv3 authentication will fail.

**Step 4 Run commit**

The configuration is committed.

----End

## Configuring OSPFv3 Area Authentication

OSPFv3 supports packet authentication, with which devices accept only the OSPFv3 packets that are authenticated. If OSPFv3 packets fail to be authenticated, OSPFv3 neighbor relationships cannot be established. Configuring OSPFv3 area authentication improves OSPFv3 network security.

## Context

With the increase in attacks on TCP/IP networks and inherent defects and flawed implementation of the TCP/IP protocol suite, the attacks have increasing impacts on the networks. Attacks on network devices may even cause a network crash or lead to network unavailability. Configuring OSPFv3 area authentication improves OSPFv3 network security. If area authentication is used, the authentication mode and password configurations on all the interfaces in the area must be identical.

 **NOTE**

By default, no authentication mode is configured for an OSPFv3 area. For security purposes, you are advised to configure an authentication mode.

OSPFv3 authentication takes effect in descending order of priority as follows: interface authentication, area authentication, and process authentication.

## Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run ospfv3 [ *process-id* ]**

The OSPFv3 view is displayed.

**Step 3 Run area *area-id***

The OSPFv3 area view is displayed.

**Step 4 Configure an authentication mode for the OSPFv3 area as required.**

- To configure the HMAC-SHA256 or HMAC-SM3 authentication mode for the OSPFv3 area, run the **authentication-mode { hmac-sha256 | hmac-sm3 }** **key-id KeyId { plain PlainText | [ cipher ] CipherText }** command.  
If you choose **plain**, the password will be saved as a cleartext in the configuration file, which provokes high security risks. To improve system security, choose ciphertext authentication and change the password periodically.
- To configure keychain authentication for the OSPFv3 area, run the **authentication-mode { keychain Keychain-Name }** command.

 **NOTE**

Before configuring keychain authentication, you need to run the **keychain** command to create a keychain, and then run the **key-id**, **key-string**, and **algorithm** commands to configure a key ID, a password, and an authentication algorithm, respectively, for this keychain. Otherwise, OSPFv3 authentication will fail.

#### Step 5 Run commit

The configuration is committed.

----End

### Configuring OSPFv3 Interface Authentication

OSPFv3 supports packet authentication, with which devices accept only the OSPFv3 packets that are authenticated. If OSPFv3 packets fail to be authenticated, OSPFv3 neighbor relationships cannot be established. Configuring OSPFv3 interface authentication improves OSPFv3 network security.

### Context

With the increase in attacks on TCP/IP networks and inherent defects and flawed implementation of the TCP/IP protocol suite, the attacks have increasing impacts on the networks. Attacks on network devices may even cause a network crash or lead to network unavailability. Configuring OSPFv3 interface authentication improves OSPFv3 network security. Interface authentication takes effect between neighboring devices' interfaces on which an authentication mode and password are configured. Interface authentication takes precedence over area authentication. For interfaces on the same subnet, the configured authentication mode and password must be identical. This requirement does not apply to the interfaces on different subnets.

 **NOTE**

By default, no authentication mode is configured for an OSPFv3 interface. For security purposes, you are advised to configure an authentication mode.

OSPFv3 authentication takes effect in descending order of priority as follows: interface authentication, area authentication, and process authentication.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run `interface interface-type interface-number`

The interface view is displayed.

### Step 3 Configure an authentication mode for the OSPFv3 interface as required.

- To configure the HMAC-SHA256 or HMAC-SM3 authentication mode for the OSPFv3 interface, run the `ospfv3 authentication-mode { hmac-sha256 | hmac-sm3 } key-id KeyId { plain PlainText | [ cipher ] CipherText } [ instance instanceld ]` command.  
If you choose **plain**, the password will be saved as a cleartext in the configuration file, which provokes high security risks. To improve system security, choose ciphertext authentication and change the password periodically.
- To configure keychain authentication for the OSPFv3 interface, run the `ospfv3 authentication-mode { keychain Keychain-Name } [ instance instanceld ]` command.

#### NOTE

Before configuring keychain authentication, you need to run the `keychain` command to create a keychain, and then run the `key-id`, `key-string`, and `algorithm` commands to configure a key ID, a password, and an authentication algorithm, respectively, for this keychain. Otherwise, OSPFv3 authentication will fail.

### Step 4 Run `commit`

The configuration is committed.

----End

## Verifying the Configuration

After configuring OSPFv3 authentication, you can check the configurations.

## Prerequisites

OSPFv3 authentication has been configured.

## Procedure

- Run the `display ospfv3 [ process-id ]` command to view the SA applied in a specified process.
- Run the `display ospfv3 [ process-id ] area [ area-id ]` command to view the SA applied in a specified area.
- Run the `display ospfv3 [ process-id ] interface [ area { area-id | area-id|ipv4 } ] [ interfaceType interfaceNum | interfaceName ]` command to check the SA applied to the interface.

----End

### 1.1.5.2.19 Configuring OSPFv3 Fast Convergence

By adjusting OSPFv3 timers, you can implement OSPF fast network convergence.

## Pre-configuration Tasks

Before configuring OSPFv3 fast convergence, complete the following tasks:

- Configure a link layer protocol.
- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPFv3 functions](#).

## Setting the Interval at Which Hello Packets Are Sent

You can adjust the value of the Hello timer to change the speed of OSPFv3 neighbor relationship establishment and the network convergence speed.

### Context

Hello packets are periodically sent to the neighbor router to detect and maintain the neighbor relationship and to elect the DR and the BDR. Based on standard protocols, the Hello timer values of neighbors must be the same. The value of the Hello timer is inversely proportional to the route convergence speed and network load.

### Procedure

#### Step 1 Run `system-view`

The system view is displayed.

#### Step 2 Run `interface interface-type interface-number`

The interface view is displayed.

#### Step 3 Run `ospfv3 timer hello interval [ conservative ] [ instance instance-id ]`

The interval at which the interface sends Hello packets is set.

The **conservative** parameter indicates that the conservative mode is enabled for the neighbor dead timer. If the conservative mode is enabled, the value configured for the dead timer using the `ospfv3 timer dead` command takes effect even when the value is less than 10s.

To speed up OSPFv3 convergence in the case of a link failure, configuring BFD for OSPFv3 is recommended. If the remote end does not support BFD for OSPFv3 or the user does not want to have BFD for OSPFv3 enabled, you are advised to specify **conservative** when running the `ospfv3 timer hello` command. In conservative mode, the value set for the dead timer using the `ospfv3 timer dead` command takes effect even if the value is less than 10 seconds; otherwise, route convergence is performed based on the OSPFv3 neighbor dead timer, which takes a long time and has a great impact on services.

#### NOTE

This interval must not be less than the time taken to perform an active/standby switchover. Otherwise, an intermittent protocol interruption may occur during a switchover. The default timer value is recommended.

#### Step 4 Run commit

The configuration is committed.

----End

### Setting the Dead Time of the Neighbor Relationship

If a device does not receive a Hello packet from its neighbor within the dead interval, the device considers the neighbor Down.

### Context

Perform the following steps on the OSPFv3 router:

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run interface *interface-type interface-number*

The interface view is displayed.

#### Step 3 Run ospfv3 timer dead *interval* [ *instance instance-id* ]

A dead timer is configured for the neighboring router.

#### NOTE

If the dead timer is shorter than 10s, the neighbor relationship may be torn down. To prevent this issue, the dead timer of 10 seconds takes effect if the value of **dead interval** is less than 10 seconds. However, if the **ospfv3 timer hello** command is run, the **conservative** parameter is specified to enable the conservative mode for the neighbor dead timer, and the configured dead timer is less than 10s, the system still uses the configured value to determine whether a neighbor is down.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring OSPFv3 Sham Hello

With OSPFv3 sham hello, device can exchange Hello and LSU and LSAck packets to maintain OSPFv3 neighbor relationships, which strengthens the neighbor detection mechanism.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run ospfv3 [ *process-id* ]

The OSPFv3 view is displayed.

**Step 3 Run sham-hello enable**

OSPFv3 sham hello is enabled.

**Step 4 Run commit**

The configuration is committed.

----End

## Setting the Interval at Which LSAs Are Updated

You can set the interval at which LSAs are updated based on network connections and router resources.

### Context

To prevent excessive consumption of network bandwidth and router resources caused by frequent network connection or route flapping, you can set the **intelligent-timer** parameter to specify the interval for updating LSAs on a stable network that requires fast route convergence. In this manner, topology or route changes can be immediately advertised to the network through LSAs, speeding up route convergence on the network.

### Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

**Step 3 Run lsa-originate-interval { 0 | intelligent-timer max-interval start-interval hold-interval [ other-type interval ] | other-type interval [ intelligent-timer max-interval start-interval hold-interval ] }**

The device is configured to set the interval for updating OSPFv3 LSAs based on the intelligent SPF timer.

**Step 4 (Optional) Run lsa-originate-interval suppress-flapping interval [ threshold count ]**

The suppression period that takes effect when sent OSPFv3 LSAs flap is configured.

If no flapping occurs among sent OSPFv3 LSAs, the configuration of the **lsa-originate-interval** command prevents the device from frequently sending LSAs. If sent OSPFv3 LSAs flap, the configuration of the **lsa-originate-interval suppress-flapping** command minimizes the impact of the flapping on services. The larger value of the two intervals is used as the suppression period.

**Step 5 Run commit**

The configuration is committed.

----End

## Setting the Interval at Which LSAs Are Received

You can set the interval at which LSAs are received based on network connections and router resources.

### Context

Setting the interval at which LSAs are received can prevent network connections or frequent route flapping from causing frequent update of LSAs. After the interval is set on a router, the router updates an LSA only after the specified interval expires.

On a stable network that requires fast route convergence, you can change the interval at which OSPFv3 LSAs are received to 0s. In this manner, the device can fast respond to topology or route changes, which speeds up route convergence.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

#### Step 3 Run **lsa-arrival-interval { interval | intelligent-timer max-interval start-interval hold-interval }**

The interval at which LSAs are received is configured.

#### Step 4 (Optional) Run **lsa-arrival-interval suppress-flapping interval [ threshold count ]**

The suppression period that takes effect when received OSPFv3 LSAs flap is configured.

If no flapping occurs among received OSPFv3 LSAs, the configuration of the **lsa-arrival-interval** command prevents the device from frequently receiving LSAs. If received OSPFv3 LSAs flap, the configuration of the **lsa-arrival-interval suppress-flapping** command minimizes the impact of the flapping on services. The larger value of the two intervals is used as the suppression period.

#### Step 5 Run **commit**

The configuration is committed.

----End

## Setting a Period During Which OSPFv3 Keeps the Maximum Cost in Local LSAs

When an OSPFv3 interface changes from down to up, if a period during which OSPFv3 keeps the maximum cost in local LSAs is configured, traffic is switched back to the recovered link only after the period elapses.

### Context

When an OSPFv3 interface changes from down to up, the OSPFv3 neighbor relationship is re-established. After OSPFv3 routes converge, traffic is switched

back. Because OSPFv3 route convergence is very fast, many services that depend on OSPFv3 routes may require delayed OSPFv3 route switchback. In this case, you can run the **ospfv3 peer hold-max-cost** command to specify a period during which OSPFv3 keeps the maximum cost in local LSAs. After the OSPFv3 neighbor relationship reaches Full state, the traffic forwarding path remains unchanged during the specified period. After this period expires, the maximum cost is restored to the original cost of the recovered link, and traffic is switched back to the recovered link.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The view of the interface that will run OSPFv3 is displayed.

### Step 3 Run **ipv6 enable**

IPv6 is enabled on the specified interface.

### Step 4 Run **ospfv3 process-id area area-id [ instance instance-id ]**

An OSPFv3 process is enabled, and the area to which the process belongs is specified.

### Step 5 Run **ospfv3 peer hold-max-cost timer hold-max-cost-value [ instance instance-id ]**

The period during which OSPFv3 keeps the maximum cost in local LSAs is set.

### Step 6 Run **commit**

The configuration is committed.

----End

## Disabling OSPFv3 LSA Aging Management

By default, the OSPFv3 LSA aging management function is enabled. To disable this function, perform this task.

## Context

LSAs are aged out if their LS age field encounters an exception, and this may cause LSA flapping or incorrect route calculation. For example, if the aging time carried in a received LSA is 2500 seconds, the device considers the LSA abnormal and reduces the aging time to 500 seconds. As a result, the LSA is aged out far sooner than expected. To address this issue, the OSPFv3 LSA aging management function is enabled by default. If the aging time in a received LSA is longer than 1800 seconds, OSPFv3 considers the LSA abnormal and changes the aging time to 1700 seconds. This operation is performed for each abnormal LSA until the aging time values of all LSAs in the area are the same. This ensures that routes can be calculated correctly.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **lsa-age refresh disable**

OSPFv3 LSA aging management is disabled.

### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring OSPFv3 fast convergence, verify brief information about OSPFv3.

## Prerequisites

OSPFv3 fast convergence has been configured.

## Procedure

- Run the **display ospfv3 [ process-id ] interface [ area area-id ] [ interface-type interface-number ]** command to view information about an OSPFv3 interface.
- Run the **display ospfv3 [ process-id ]** command to check brief OSPFv3 information.

----End

### 1.1.5.2.20 Configuring the OSPFv3 GR Helper

To avoid traffic interruption and route flapping caused by the active/standby switchover, you can enable OSPFv3 GR.

## Context

GR ensures normal traffic forwarding and non-stop forwarding of key services during the restart of routing protocols. GR is one of high availability (HA) technologies. HA technologies comprise a set of comprehensive techniques, such as fault-tolerant redundancy, link protection, faulty node recovery, and traffic engineering. As a fault-tolerant redundancy technology, GR is widely used to ensure non-stop forwarding of key services during master/slave main control board switchover and system upgrade.

#### NOTE

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports only the GR Helper.

## Pre-configuration Tasks

Before configuring OSPFv3 GR, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- **Configure basic OSPFv3 functions.**

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 process-id**

The OSPFv3 view is displayed.

### Step 3 Configure a basic ACL:

1. Run **helper-role [ { ip-prefix ip-prefix-name | acl-number acl-number | acl-name acl-name } | max-grace-period period | planned-only | lsa-checking-ignore ]**\*

OSPFv3 GR is enabled.

2. Run **quit**

Return to the system view.

3. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

4. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ]**\*

A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the following command to check the preceding configuration:

- Run the **display ospfv3 [ process-id ] graceful-restart-information** command to view the OSPFv3 GR Helper status.

### 1.1.5.2.21 Configuring OSPFv3 to Report Network Topology Information to BGP-LS

After OSPFv3 topology information is advertised to BGP-LS, BGP-LS reports the information to a controller, which implements path planning.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **ospfv3 [ process-id ]**

OSPFv3 is enabled, and the OSPFv3 view is displayed.

##### Step 3 Run **bgp-ls enable**

The OSPFv3 process is enabled to advertise topology information.

##### Step 4 Run **bgp-ls identifier [ identifier-value ]**

OSPFv3 topology advertisement is configured, and a topology ID is specified.

##### Step 5 Run **commit**

The configuration is committed.

----End

### 1.1.5.2.22 Improving OSPFv3 Network Stability

A stable OSPFv3 network features less route flapping, normal router performance, and high network performance.

#### Usage Scenario

By setting timers, you can reduce the number of unnecessary packets on networks and the load on the device, and improve network performance.

By adjusting the SPF calculation interval, you can mitigate resource consumption due to frequent network changes.

#### Pre-configuration Tasks

Before improving the security of an OSPFv3 network, complete the following tasks:

- Configure a link layer protocol.
- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic OSPF3 functions](#).

#### Setting the OSPFv3 Preference

If multiple dynamic routing protocols run on a device, the device needs to select optimal routes from the routes of these protocols. In this case, you can set a priority for each routing protocol. In this manner, when different protocols discover

the routes to the same destination, the route discovered by the protocol with the highest priority is selected.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

**Step 3** Run **preference [ ase ] { preference | route-policy route-policy-name | route-filter route-filter-name } \***

A preference is set for OSPFv3.

**Step 4** Run **commit**

The configuration is committed.

----End

## Setting the Delay in Transmitting LSAs on the Interface

It takes time to transmit OSPFv3 packets on a link. Therefore, a certain delay is added to the aging time of an LSA before the LSA is sent. This configuration is especially necessary on low-speed links.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface interface-type interface-number**

The OSPFv3 interface view is displayed.

**Step 3** Run **ospfv3 trans-delay interval [ instance instance-id ]**

The delay in transmitting LSAs is set on the interface.

**Step 4** Run **commit**

The configuration is committed.

----End

## Setting the Interval at Which LSAs Are Retransmitted Between Adjacent Routers

You can set an appropriate interval at which LSAs are retransmitted based on network conditions in order to accelerate convergence.

## Context

After sending an LSA to its neighbor, a device waits for a response. If the device does not receive an acknowledgement packet within the Nth LSA retransmission interval, the device retransmits the LSA to the neighbor.

First LSA retransmission interval = Configured interval at which LSAs are retransmitted, which is *interval*

Second LSA retransmission interval = Configured interval at which LSAs are retransmitted, which is *interval*

Third LSA retransmission interval = Configured interval at which LSAs are retransmitted, which is *interval*

Fourth LSA retransmission interval = Configured interval at which LSAs are retransmitted (*interval*) x 2

Fifth LSA retransmission interval = Configured interval at which LSAs are retransmitted (*interval*) x 2<sup>2</sup>

... ...

Nth LSA Retransmission Interval = Configured interval at which LSAs are retransmitted (*interval*) x 2^(n-3).

If *interval* x 2^(n-3) is greater than 30, the Nth LSA retransmission interval is 30.

If the configured interval at which LSAs are retransmitted (*interval*) is greater than 30, the Nth LSA retransmission interval equals *interval*.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The OSPFv3 interface view is displayed.

### Step 3 Run **ospfv3 timer retransmit interval [ instance instance-id ]**

An interval for retransmitting LSAs to the adjacent device is configured.



Set the LSA retransmission interval to a proper value. An excessively short interval will cause unnecessary retransmission. Generally, the interval should be longer than the time taken for round-trip packet transmission between two ends.

### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring a Route Calculation Delay to Suppress Frequent LSA Flapping

### Context

Frequent OSPFv3 LSA flapping may lead to route flapping, adversely affecting services. To address this problem, run the **maxage-lsa route-calculate-delay** command to configure the device to delay route calculation when it receives a MaxAge Router LSA, which suppresses the frequent OSPFv3 LSA flapping that may occur.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

### Step 3 Run **maxage-lsa route-calculate-delay delay-interval**

A route calculation delay is configured and will be triggered when the device receives a MaxAge Router LSA.

### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring the SPF Timer

By setting the interval for Shortest Path First (SPF) calculation, you can reduce resource consumption caused by frequent network changes.

## Context

Whenever the Link-state Database (LSDB) of OSPFv3 changes, the shortest path should be recalculated. Calculating the shortest path each time the LSDB changes consumes enormous resources and lowers the efficiency of the device. In this case, you can adjust values of *delay-interval* and *hold-interval* to prevent bandwidth exhaustion and route consumption caused by frequent network changes. You can use the **intelligent-timer** parameter to configure the device to perform SPF calculation at an interval (at the millisecond level) based on an intelligent timer. This speeds up network convergence.

## Procedure

- Configure an SPF normal timer.

- Run **system-view**

The system view is displayed.

- Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

- Run **spf timers delay-interval hold-interval**

An SPF normal timer is configured.

- Run **commit**

The configuration is committed.

- Configure an SPF intelligent timer.

- Run **system-view**

The system view is displayed.

b. Run **ospfv3** [*process-id*]

The OSPFv3 view is displayed.

c. Run **spf-schedule-interval** { *delay-interval hold-interval* | **intelligent-timer** *max-interval start-interval hold-interval-1 millisecond interval2* }

The SPF intelligent timer is set.

The interval for SPF calculation based on the intelligent timer is described as follows:

- i. The initial interval for SPF calculation is specified by *start-interval*.
- ii. The interval for SPF calculation for the nth ( $n \geq 2$ ) time equals *hold-interval-1*  $\times 2^{(n - 2)}$ .
- iii. After the interval specified by *hold-interval-1*  $\times 2^{(n - 2)}$  reaches the maximum interval specified by *max-interval*, OSPFv3 keeps using the maximum interval for SPF calculation.
- iv. If no flapping occurs during the interval from the end of the last SPF calculation to the start of the next SPF calculation, and the interval exceeds the maximum interval specified by *max-interval*, the intelligent timer exits.
- v. If no flapping occurs in the previous interval but occurs in the current interval, SPF calculation is delayed for a period of *start-interval*. After the SPF calculation is complete, the current interval will be applied when waiting for the next SPF calculation.

 NOTE

The normal SPF timer and intelligent SPF timer are mutually exclusive.

d. Run **commit**

The configuration is committed.

----End

## Configuring a Stub Router

When the device is overloaded and cannot forward any other packets, you can configure it as a stub router. After the router is configured as a stub router, other OSPFv3 devices do not use this device to forward data, but they can have a route to this stub router.

## Context

A stub router controls traffic and instruct other OSPFv3 devices not to use it to forward data. Other OSPFv3 devices can have a route to the stub router.

The link cost in the Router LSAs generated by the stub router is set to the maximum value (65535).

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

**Step 2** Run **ospfv3 [ process-id ]**

The OSPFv3 view is displayed.

**Step 3** Run **stub-router [[ on-startup [ interval ] ] | [ intra-area-prefix-lsa ] | [ external-lsa [ externallsa-metric ] ] | [ inter-area-prefix-lsa [ interareaprefixlsa\_metric ] ] ]\***

A stub router is configured.

The parameter **on-startup [ interval ]** specifies the interval during which the device remains a stub router.

 **NOTE**

The stub router configured through this command has nothing to do with the routers in a stub area.

**Step 4** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After improving the OSPFv3 network stability, verify brief information about OSPFv3 and the IP routing table.

## Procedure

- Run the **display ospfv3 [ process-id ] interface [ no-peer | area area-id ] [ interface-type interface-number ]** command to view information about an OSPFv3 interface.
- Run the **display ospfv3 [ process-id ] routing** command to check information about the OSPFv3 routing table.
- Run the **display ospfv3 [ process-id ] statistics maxage-lsa** command to check information about MaxAge Router-LSAs.

----End

### 1.1.5.2.23 Configuring the Network Management Function of OSPFv3

OSPFv3 supports the network management function. You can configure the OSPFv3 Management Information Base (MIB) and bind it to an OSPFv3 process through Simple Network Management Protocol (SNMP). In this manner, the OSPFv3 MIB manages multicast information exchanged between the Network Management Station (NMS) and agents.

## Pre-configuration Tasks

Before configuring the network management function of OSPFv3, complete the following tasks:

- Configure a link layer protocol.
- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.

- **Configure basic OSPFv3 functions.**

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ospfv3 mib-binding process-id**

The OSPFv3 process is bound to the MIB.

### Step 3 Run **commit**

The configuration is committed.

----End

## Checking the Configurations

Run the following command to check the previous configuration:

- Run the **display current-configuration** command to check whether an OSPFv3 process has been bound to the OSPFv3 MIB.

### 1.1.5.2.24 Whitelist Session-CAR for OSPFv3

You can configure whitelist session-CAR for OSPFv3 to isolate bandwidth resources by session for OSPFv3 packets. This configuration prevents bandwidth preemption among OSPFv3 sessions in the case of a traffic burst.

## Context

When OSPFv3 packets suffer a traffic burst, bandwidth may be preempted among different OSPFv3 sessions. To resolve this problem, you can configure whitelist session-CAR for OSPFv3 to isolate bandwidth resources by session. If the default parameters of whitelist session-CAR for OSPFv3 do not meet service requirements, you can adjust them as required.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **whitelist session-car ospfv3 { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } \***

Parameters of whitelist session-CAR for OSPFv3 are configured.

In normal cases, you are advised to use the default values of these parameters.

### Step 3 (Optional) Run **whitelist session-car ospfv3 disable**

Whitelist session-CAR for OSPFv3 is disabled.

By default, whitelist session-CAR for OSPFv3 is enabled. In normal cases, you are advised to keep this function enabled. Disable it if it becomes abnormal or adversely affects other services.

#### Step 4 Run commit

The configuration is committed.

----End

### Verifying the Configuration

After configuring whitelist session-CAR for OSPFv3, verify the configuration.

- Run the **display cpu-defend whitelist-v6 session-car ospfv3 statistics slot slot-id** command to check statistics about whitelist session-CAR for OSPFv3 on a specified interface board.

#### 1.1.5.2.25 Configuring Micro-Isolation CAR for OSPFv3

By default, micro-isolation CAR is enabled for OSPFv3, implementing micro-isolation protection for OSPFv3 packets based on interfaces and destination IP addresses to protect session establishment.

### Context

By default, micro-isolation CAR is enabled for OSPFv3, implementing micro-isolation protection for OSPFv3 packets based on interfaces and destination IP addresses to protect session establishment. When OSPFv3 packets encounter a traffic burst, a large number of OSPFv3 packets may preempt interface bandwidth resources. In the case of an attack, a large number of invalid packets sent to other IP addresses may preempt the bandwidth. Therefore, you are advised not to disable this function.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run **micro-isolation protocol-car ospfv3 { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value }\***

Micro-isolation CAR parameters are configured for OSPFv3.

In normal cases, you are advised to use the default values of these parameters. *pir-value* must be greater than or equal to *cir-value*, and *pbs-value* must be greater than or equal to *cbs-value*.

#### Step 3 (Optional) Run **micro-isolation protocol-car ospfv3 disable**

Micro-isolation CAR is disabled for OSPFv3.

By default, micro-isolation CAR for OSPFv3 is enabled. To disable this function, run the **micro-isolation protocol-car ospfv3 disable** command. If this function is disabled, micro-isolation protection is no longer implemented for OSPFv3 packets based on interfaces and destination IP addresses. In normal cases, you are advised to keep micro-isolation CAR enabled for OSPFv3.

#### Step 4 Run commit

The configuration is committed.

----End

### 1.1.5.2.26 Configuring the Threshold Alarm Function for the Number of OSPFv3 Neighbors

If the alarm function is configured for the number of OSPFv3 neighbors and the number of OSPFv3 neighbors exceeds the threshold, an alarm is generated.

#### Context

When the number of established OSPFv3 neighbor relationships exceeds the specifications of the device, establishing more OSPFv3 neighbor relationships may cause the neighbor status to be unstable. To prevent this problem, you can enable the alarm function for the number of OSPFv3 neighbor relationships so that an alarm is generated when the number of OSPFv3 neighbor relationships exceeds the upper alarm threshold.

#### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **ospfv3 peer alarm-threshold *threshold-value upper-limit upper-value lower-limit lower-value***

The threshold alarm function is configured for the number of OSPFv3 neighbors.

**Step 3** Run **commit**

The configuration is committed.

----End

### 1.1.5.2.27 Maintaining OSPFv3

Maintaining OSPFv3 involves resetting OSPFv3 and clearing OSPFv3 statistics.

#### Clearing OSPFv3 Statistics

This section describes how to clear OSPFv3 statistics, including statistics about OSPFv3 counters and statistics about session-CAR.

#### Context

**NOTICE**

OSPFv3 information cannot be restored after you clear it. Exercise caution when clearing it.

To clear OSPFv3 information, run the following reset commands in the user view.

## Procedure

- Run the **reset ospfv3 { process-id| all } counters [ neighbor [ interface-type interface-number ] [ nbrrouter-id ] ]** command to reset OSPFv3 counters.
- Run the **reset ospfv3 { process-id| all } counters maxage-lsa** command to delete the statistics about router-LSAs that have reached the aging time.
- Run the **reset ospfv3 { process-id| all } frr** command to perform OSPFv3 IP FRR calculation again.
- Run the **reset ospfv3 { process-id| all } peer [ interface-type interface-number ] router-id** command to restart an OSPFv3 neighbor.
- Run the **reset cpu-defend whitelist-v6 session-car ospfv3 statistics slot slot-id** command to clear statistics about whitelist session-CAR for OSPFv3 on a specified interface board.

----End

## Resetting OSPFv3

Restarting OSPFv3 can reset OSPFv3.

## Context

### NOTICE

Resetting an OSPFv3 connection interrupts the OSPFv3 adjacency between devices. Therefore, exercise caution when resetting an OSPFv3 connection.

After modifying the OSPFv3 routing policy or protocol, reset the OSPFv3 connection to validate the modification. To reset OSPFv3 connections, run the following commands as required in the user view.

## Procedure

- Run the **reset ospfv3 { process-id| all } command** to restart the OSPFv3 process.
- Run the **reset ospfv3 { process-id| all } peer [ interface-type interface-number ] router-id** command to restart an OSPFv3 neighbor.
- Run the **reset ospfv3 { process-id| all } redistribution** command to import routes again.

----End

## Disabling OSPFv3 Memory Overload Control

The OSPFv3 memory overload control function is enabled by default. You can disable this function if necessary.

## Context

By default, OSPFv3 memory overload control is enabled. When the system memory is overloaded, each module needs to take necessary measures to control

the memory usage. Upon receiving a memory overload notification from the system, the OSPFv3 module no longer imports new routes and starts to control neighbor relationship establishment based on the memory overload condition to enhance OSPFv3 resilience. In this case, new neighbor relationships cannot be established. For existing neighbor relationships, if they are in the Full state, they will be retained; if they are in a non-Full state, they are not reestablished until the memory recovers from overload.

## Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** Run the **ospf memory-overload control disable** command to disable OSPFv3 memory overload control.



To reduce the impact of memory overload on services, you are advised not to disable OSPFv3 memory overload control.

- Step 3** Run the **commit** command to commit the configuration.

----End

## Disabling OSPFv3 CPU Overload Control

The OSPFv3 CPU overload control function is enabled by default. You can disable this function if necessary.

## Context

By default, OSPFv3 CPU overload control is enabled. If a device's CPU is overloaded, each module takes necessary measures to control its own CPU usage accordingly. Upon receiving a CPU overload notification from the system, the OSPFv3 module controls the speeds of some internal computing processes and the establishment of neighbor relationships based on the CPU overload condition to enhance the resilience of OSPFv3. In this case, new neighbor relationships cannot be established. For original neighbor relationships, if a neighbor relationship is in the Full state, it will be retained; if a neighbor relationship is in a non-Full state, establishment of the neighbor relationship is paused and can continue only after the CPU recovers from overload.

## Procedure

- Step 1** Run **system-view**

The system view is displayed.

- Step 2** Run **ospf cpu-overload control disable**

OSPFv3 CPU overload control is disabled.



To minimize the impact of CPU overload upon services, you are advised not to disable OSPFv3 CPU overload control.

### Step 3 Run commit

The configuration is committed.

----End

## 1.1.5.2.28 Configuration Examples for OSPFv3

This section provides several OSPFv3 configuration examples.

### Example for Configuring Basic OSPFv3 Functions

This section describes how to configure basic OSPFv3 functions, including enabling OSPF on each router and specifying network segments in different areas.

### Networking Requirements

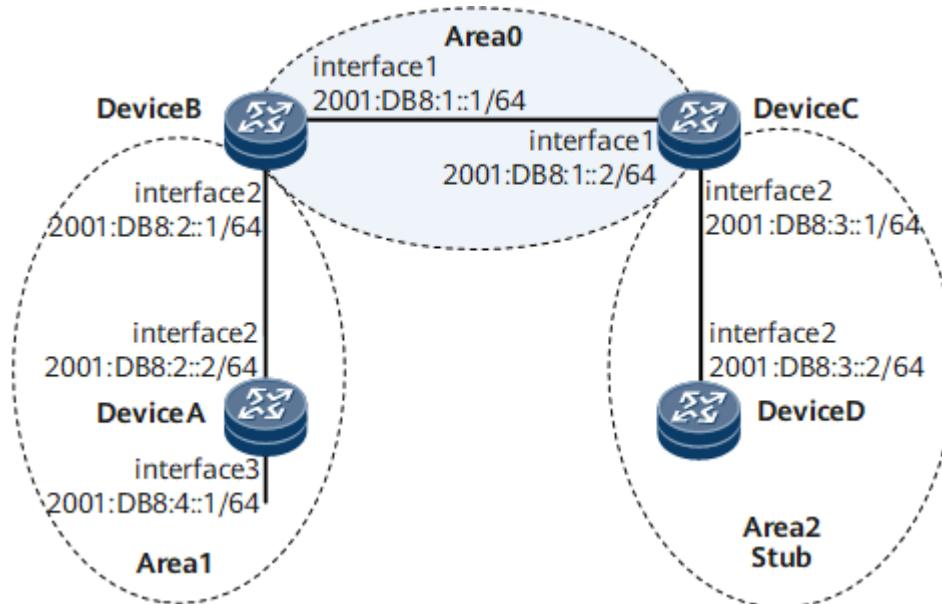
On the network shown in [Figure 1-188](#), all routers run OSPFv3. The entire AS is divided into three areas. DeviceB and DeviceC serve as ABRs to forward inter-area routes.

After the configuration is complete, each router should learn the routes to all network segments in the AS.

**Figure 1-188** Networking for configuring OSPFv3 areas



Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



### Configuration Notes

When configuring basic OSPFv3 functions, note the following rules:

- The backbone area is responsible for forwarding inter-area routes. In addition, the routing information between non-backbone areas must be forwarded through the backbone area. OSPFv3 defines the following rules for the backbone area:
  - Connectivity must be available between non-backbone areas and the backbone area.
  - Connectivity must be available over the backbone area.
- The intervals at which Hello, Dead, and Poll packets are sent on the local interface must be the same as those on the remote interface. Otherwise, the OSPFv3 neighbor relationship cannot be established.
- To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication." OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable basic OSPFv3 functions on each router.
2. Specify network segments in different areas.

## Data Preparation

To complete the configuration, you need the following data:

| Device Name | Router ID | Process ID | IPv6 Address                                                   |
|-------------|-----------|------------|----------------------------------------------------------------|
| Device A    | 1.1.1.1   | 1          | Area 1:<br>2001:DB8:4::1/64<br>and<br>2001:DB8:2::2/64         |
| Device B    | 2.2.2.2   | 1          | Area 0:<br>2001:DB8:1::1/64<br><br>Area 1:<br>2001:DB8:2::1/64 |
| Device C    | 3.3.3.3   | 1          | Area 0:<br>2001:DB8:1::2/64<br><br>Area 2:<br>2001:DB8:3::1/64 |
| Device D    | 4.4.4.4   | 1          | Area 2:<br>2001:DB8:3::2/64                                    |

## Procedure

**Step 1** Assign an IPv6 address to each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic OSPFv3 functions.

# Configure DeviceA.

```
[~DeviceA] ospfv3
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] area 0.0.0.1
[*DeviceA-ospfv3-1] quit
[*DeviceA] interface gigabitethernet3/0/0
[*DeviceA-GigabitEthernet3/0/0] ospfv3 1 area 1
[*DeviceA-GigabitEthernet3/0/0] quit
[*DeviceA] interface gigabitethernet2/0/0
[*DeviceA-GigabitEthernet2/0/0] ospfv3 1 area 1
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[~DeviceB] ospfv3
[*DeviceB-ospfv3-1] router-id 2.2.2.2
[*DeviceB-ospfv3-1] area 0.0.0.0
[*DeviceB-ospfv3-1] area 0.0.0.1
[*DeviceB-ospfv3-1] quit
[*DeviceB] interface gigabitethernet1/0/0
[*DeviceB-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet2/0/0
[*DeviceB-GigabitEthernet2/0/0] ospfv3 1 area 1
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
[~DeviceC] ospfv3
[*DeviceC-ospfv3-1] router-id 3.3.3.3
[*DeviceC-ospfv3-1] area 0.0.0.0
[*DeviceC-ospfv3-1] area 0.0.0.2
[*DeviceC-ospfv3-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ospfv3 1 area 2
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] ospfv3
[*DeviceD-ospfv3-1] router-id 4.4.4.4
[*DeviceD-ospfv3-1] area 0.0.0.2
[*DeviceD-ospfv3-1] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] ospfv3 1 area 2
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] commit
```

**Step 3** Verify the configuration.

# Display the OSPFv3 neighbors of DeviceB. The state Full indicates that a neighbor relationship is established.

```
[*DeviceB] display ospfv3 peer
```

```
OSPFv3 Process (1)
Total number of peer(s): 2
Peer(s) in full state: 2
OSPFv3 Area (0.0.0.1)
Neighbor ID Pri State Dead Time Interface Instance ID
1.1.1.1 1 Full/ - 00:00:34 GigabitEthernet2/0/0 0

OSPFv3 Area (0.0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
3.3.3.3 1 Full/ - 00:00:32 GigabitEthernet1/0/0 0
```

# Display the OSPFv3 neighbors of DeviceC.

```
[*DeviceC] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 2
Peer(s) in full state: 2
OSPFv3 Area (0.0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
2.2.2.2 1 Full/ - 00:00:37 GigabitEthernet1/0/0 0

OSPFv3 Area (0.0.0.2)
Neighbor ID Pri State Dead Time Interface Instance ID
4.4.4.4 1 Full/ - 00:00:33 GigabitEthernet2/0/0 0
```

# Display the OSPFv3 routing table of DeviceD.

```
[~DeviceD] display ospfv3 routing
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
 N - NSSA
Flags: A - Added to URT6, LT - Locator Routing

OSPFv3 Process (1)
Destination Metric
 Next-hop
 IA 2001:DB8:1::/64 2
 via FE80::1572:0:5EF4:1, GigabitEthernet2/0/0
 IA 2001:DB8:2::/64 3
 via FE80::1572:0:5EF4:1, GigabitEthernet2/0/0
 2001:DB8:3::/64 1
 directly-connected, GigabitEthernet2/0/0
 IA 2001:DB8:4::/64 4
 via FE80::1572:0:5EF4:1, GigabitEthernet2/0/0
```

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
ospfv3 1 area 0.0.0.1
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
ospfv3 1 area 0.0.0.1
#
ospfv3 1
router-id 1.1.1.1
area 0.0.0.1
#
```

- ```
return
● DeviceB configuration file
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
ospfv3 1 area 0.0.0.1
#
ospfv3 1
router-id 2.2.2.2
area 0.0.0.0
area 0.0.0.1
#
return

● DeviceC configuration file
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
ospfv3 1 area 0.0.0.2
#
ospfv3 1
router-id 3.3.3.3
area 0.0.0.0
area 0.0.0.2
#
return

● DeviceD configuration file
#
sysname DeviceD
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::2/64
ospfv3 1 area 0.0.0.2
#
ospfv3 1
router-id 4.4.4.4
area 0.0.0.2
#
return
```

Example for Configuring OSPFv3 DR Election

This section describes how to set the DR priority on an interface for DR election on a broadcast network.

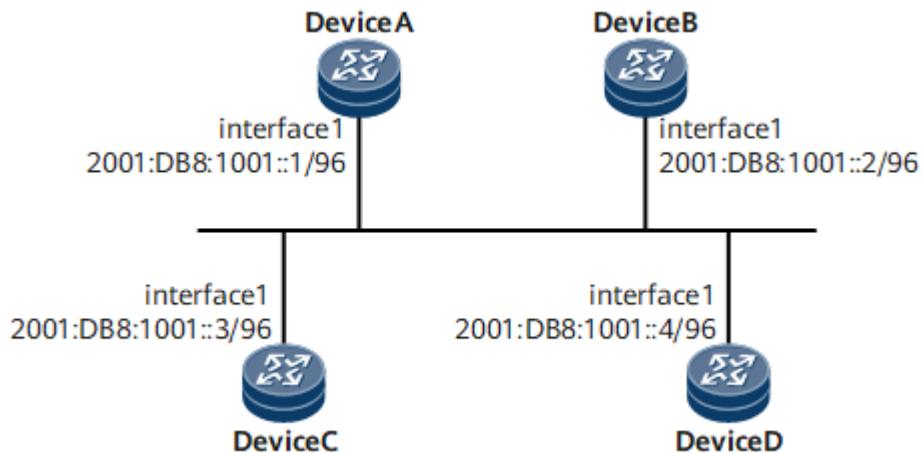
Networking Requirements

On the network shown in [Figure 1-189](#), DeviceA has the highest priority (100) on the network and is elected as the DR; DeviceC has the second highest priority and is elected as the BDR. DeviceB has the priority of 0 and cannot be elected as a DR or a BDR; no priority is configured for DeviceD, and therefore, and DeviceD uses the default value (1).

Figure 1-189 Networking for configuring OSPFv3 DR election

 NOTE

Interface 1 in this example represents GE 1/0/0.



Configuration Notes

Reconfiguring the DR priority for the router does not change the DR or BDR on a network. You can use either of the following methods to re-elect a DR or BDR. However, the following methods will disconnect OSPFv3 adjacencies and therefore are not recommended.

- Restart the OSPFv3 processes on all routers.
- Configure the **shutdown** and **undo shutdown** commands on the interfaces where the OSPFv3 neighbor relationships are established.

To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

Configuration Roadmap

1. Configure the router ID on each router, enable OSPFv3, and specify the network segment.
2. Check the DR/BDR status with the default priority.
3. Configure the DR priority on the interface and check the DR/BDR status.

Data Preparation

To complete the configuration, you need the following data:

- Router ID (1.1.1.1) and DR priority (100) of Device A
- Router ID (2.2.2.2) and DR priority (0) of Device B
- Router ID (3.3.3.3) and DR priority (2) of Device C
- Router ID (4.4.4.4) and DR priority (1, default value) of Device D

Procedure

Step 1 Assign an IPv6 address to each interface.

For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic OSPFv3 functions.

Configure DeviceA, enable OSPFv3, and set its router ID to 1.1.1.1.

```
[*DeviceA] ospfv3
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] commit
```

Configure DeviceB, enable OSPFv3, and set its router ID to 2.2.2.2.

```
[*DeviceB] ospfv3
[*DeviceB-ospfv3-1] router-id 2.2.2.2
[*DeviceB-ospfv3-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] commit
```

Configure DeviceC, enable OSPFv3, and set its router ID to 3.3.3.3.

```
[*DeviceC] ospfv3
[*DeviceC-ospfv3-1] router-id 3.3.3.3
[*DeviceC-ospfv3-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] commit
```

Configure DeviceD, enable OSPFv3, and set its router ID to 4.4.4.4.

```
[*DeviceD] ospfv3
[*DeviceD-ospfv3-1] router-id 4.4.4.4
[*DeviceD-ospfv3-1] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] commit
```

Display the neighbors of DeviceA. You can view the DR priority and the neighbor status. By default, the DR priority is 1. DeviceD is the DR, and DeviceC is the BDR.

NOTE

The router with the greatest router ID is the DR when routers have the same priority. If an Ethernet interface of the router becomes a DR, the other broadcast interfaces of the router have the highest priority in DR election. That is, the DR cannot be preempted.

```
[~DeviceA] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 3
Peer(s) in full state: 2
OSPFv3 Area (0.0.0.0)
Neighbor ID   Pri  State          Dead Time  Interface  Instance ID
2.2.2.2      1    2-Way/DROther  00:00:32   GE1/0/0      0
3.3.3.3      1    Full/Backup    00:00:36   GE1/0/0      0
4.4.4.4      1    Full/DR       00:00:38   GE1/0/0      0
```

Display the neighbors of DeviceD. The command output shows that all neighbors of DeviceD are in the Full state.

```
[~DeviceD] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 3
Peer(s) in full state: 3
OSPFv3 Area (0.0.0.0)
Neighbor ID   Pri  State          Dead Time  Interface  Instance ID
1.1.1.1      1    Full/DROther  00:00:32   GE1/0/0      0
2.2.2.2      1    Full/DROther  00:00:35   GE1/0/0      0
3.3.3.3      1    Full/Backup    00:00:30   GE1/0/0      0
```

Step 3 Set the DR priority of the interface.

Set the DR priority of DeviceA to 100.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ospfv3 dr-priority 100
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] commit
```

Set the DR priority of DeviceB to 0.

```
[~DeviceB] interface gigabitethernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] ospfv3 dr-priority 0
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] commit
```

Set the DR priority of DeviceC to 2.

```
[~DeviceC] interface gigabitethernet 1/0/0
[~DeviceC-GigabitEthernet1/0/0] ospfv3 dr-priority 2
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] commit
```

Display the neighbors of DeviceA. The command output shows that the DR priority is updated and that the DR and BDR remain unchanged.

```
[~DeviceA] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 2
Peer(s) in full state: 3
OSPFv3 Area (0.0.0.0)
Neighbor ID   Pri  State          Dead Time  Interface  Instance ID
2.2.2.2      0    2-Way/DROther  00:00:34   GE1/0/0      0
3.3.3.3      2    Full/Backup    00:00:38   GE1/0/0      0
4.4.4.4      1    Full/DR       00:00:31   GE1/0/0      0
```

Display the neighbors of DeviceD. The command output shows that DeviceD remains the DR.

```
[~DeviceD] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 3
Peer(s) in full state: 3
OSPFv3 Area (0.0.0.0)
Neighbor ID   Pri  State          Dead Time  Interface  Instance ID
1.1.1.1      100  Full/DROther  00:00:36   GE1/0/0      0
```

2.2.2.2	0	Full/DROther	00:00:30	GE1/0/0	0
3.3.3.3	2	Full/Backup	00:00:36	GE1/0/0	0

Step 4 Re-elect the DR/BDR.

Restart all routers (or run the **shutdown** and **undo shutdown** commands on the interface that establishes the OSPFv3 neighbor relationship), and re-elect the DR/BDR.

Step 5 Verify the configuration.

Display the neighbors of DeviceA. The command output shows that DeviceC is the BDR.

```
[~DeviceA] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 3
Peer(s) in full state: 3
OSPFv3 Area (0.0.0.0)
Neighbor ID   Pri  State        Dead Time  Interface Instance ID
2.2.2.2       0    Full/DROther  00:00:31   GE1/0/0   0
3.3.3.3       2    Full/Backup   00:00:36   GE1/0/0   0
4.4.4.4       1    Full/DROther  00:00:39   GE1/0/0   0
```

Check the neighbors of DeviceD. The command output shows that DeviceA functions as the DR.

```
[~DeviceD] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 3
Peer(s) in full state: 2
OSPFv3 Area (0.0.0.0)
Neighbor ID   Pri  State        Dead Time  Interface Instance ID
1.1.1.1       100  Full/DR      00:00:39   GE1/0/0   0
2.2.2.2       0    2-Way/DROther 00:00:35   GE1/0/0   0
3.3.3.3       2    Full/Backup   00:00:39   GE1/0/0   0
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1001::1/96
ospfv3 1 area 0.0.0.0
ospfv3 dr-priority 100
#
ospfv3 1
router-id 1.1.1.1
area 0.0.0.0
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1001::2/96
ospfv3 1 area 0.0.0.0
```

- ```
ospfv3 dr-priority 0
#
ospfv3 1
router-id 2.2.2.2
area 0.0.0.0
#
return

● DeviceC configuration file

#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1001::3/96
ospfv3 1 area 0.0.0.0
ospfv3 dr-priority 2
#
ospfv3 1
router-id 3.3.3.3
area 0.0.0.0
#
return

● DeviceD configuration file

#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1001::4/96
ospfv3 1 area 0.0.0.0
#
ospfv3 1
router-id 4.4.4.4
area 0.0.0.0
#
return
```

## Example for Configuring OSPFv3 IP FRR

This section describes the procedure for configuring OSPFv3 IP FRR, including how to block FRR on certain interfaces to prevent the links connected to these interfaces from functioning as backup links and how to bind OSPFv3 IP FRR to a BFD session.

## Networking Requirements

When a fault occurs on the network, OSPFv3 IP FRR rapidly switches traffic to the backup link without waiting for route convergence. This ensures non-stop traffic forwarding.

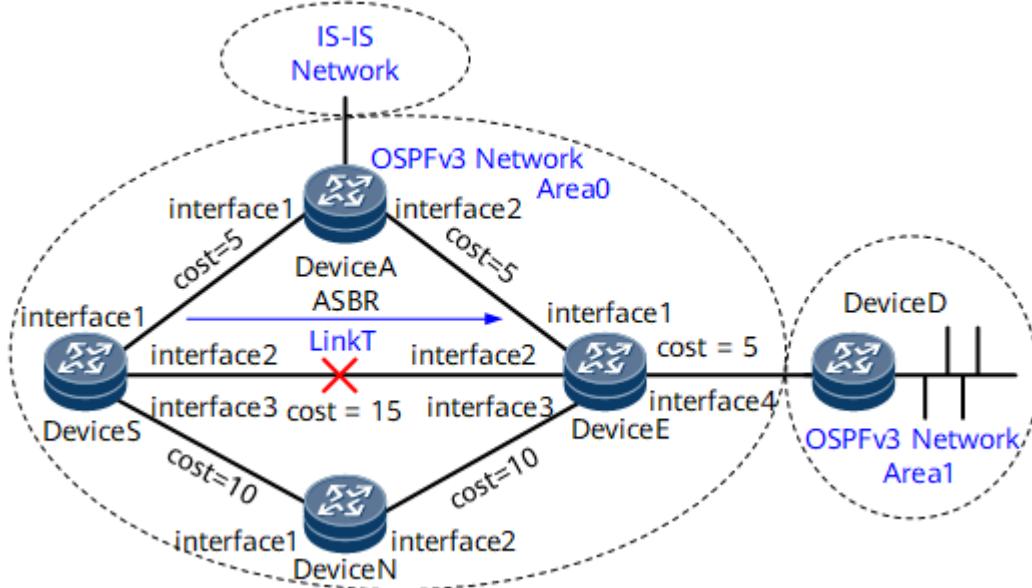
As shown in [Figure 1-190](#):

- OSPFv3 runs on all devices.
- The link cost meets the OSPFv3 IP FRR traffic protection inequality.
- If the primary link T fails, it is required that the traffic from Device S be rapidly redirected to the backup link that passes through Device N.
- Based on the network planning, the link passing through Device A does not function as a backup link.

Figure 1-190 Networking for configuring OSPFv3 IP FRR

 NOTE

Interfaces 1 through 4 in this example represent GE1/0/0, GE2/0/0, GE3/0/0, and GE1/0/1, respectively.



## Configuration Notes

When configuring OSPFv3 IP FRR, note the following points:

- Before configuring OSPFv3 IP FRR, block FRR on certain interfaces to prevent the links connected to these interfaces from functioning as backup links. After that, the link where the interface resides is not calculated as a backup link during FRR calculation.
- When OSPFv3 IP FRR is configured, the underlying layer must be able to quickly respond to link changes so that traffic can be quickly switched to the backup link. After the **bfd all-interfaces frr-binding** command is run, the BFD session status is bound to the link status of the interface (when the BFD session goes down, the link status of the interface also goes down). In this manner, faults can be rapidly detected.
- To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPFv3 functions on each router. (For configuration details, see [Example for Configuring Basic OSPFv3 Functions](#).)
2. Configure BFD for OSPFv3 on all devices in Area 0.
3. Set the costs of links to ensure that link T is selected to transmit traffic.
4. Block FRR on the specified interface on Device S.

5. Enable OSPFv3 IP FRR on DeviceA to protect the traffic forwarded by DeviceA.

## Data Preparation

To complete the configuration, you need the following data:

| Device   | Router ID | Interface | IPv6 Address        |
|----------|-----------|-----------|---------------------|
| Device S | 1.1.1.1   | GE1/0/0   | 2001:DB8:1000::1/96 |
|          |           | GE2/0/0   | 2001:DB8:1001::1/96 |
|          |           | GE3/0/0   | 2001:DB8:1002::1/96 |
| Device A | 2.2.2.2   | GE1/0/0   | 2001:DB8:1000::2/96 |
|          |           | GE2/0/0   | 2001:DB8:2000::2/96 |
| Device N | 3.3.3.3   | GE1/0/0   | 2001:DB8:1002::2/96 |
|          |           | GE2/0/0   | 2001:DB8:2002::2/96 |
| Device E | 4.4.4.4   | GE1/0/0   | 2001:DB8:2000::1/96 |
|          |           | GE2/0/0   | 2001:DB8:2001::1/96 |
|          |           | GE3/0/0   | 2001:DB8:2002::1/96 |
|          |           | GE1/0/1   | 2001:DB8:3000::1/96 |

## Procedure

- Step 1** Assign an IPv6 address to each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic OSPFv3 functions. See [Example for Configuring Basic OSPFv3 Functions](#).
- Step 3** Configure BFD for OSPFv3 on all devices in Area 0. See [Example for Configuring BFD for OSPFv3](#).
- Step 4** Set the costs of links to ensure that link T is selected to transmit traffic.

# Configure Device S.

```
[~DeviceS] interface gigabitethernet1/0/0
```

```
[~DeviceS-GigabitEthernet1/0/0] ospfv3 cost 5
[*DeviceS-GigabitEthernet1/0/0] quit
[*DeviceS] interface gigabitethernet2/0/0
[~DeviceS-GigabitEthernet2/0/0] ospfv3 cost 15
[*DeviceS-GigabitEthernet2/0/0] quit
[*DeviceS] interface gigabitethernet3/0/0
[~DeviceS-GigabitEthernet3/0/0] ospfv3 cost 10
[*DeviceS-GigabitEthernet3/0/0] quit
[*DeviceS] commit
```

# Configure Device A.

```
[~DeviceA] interface gigabitethernet1/0/0
[~DeviceA-GigabitEthernet1/0/0] ospfv3 cost 5
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet2/0/0
[~DeviceA-GigabitEthernet2/0/0] ospfv3 cost 5
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

# Configure Device N.

```
[~DeviceN] interface gigabitethernet1/0/0
[~DeviceN-GigabitEthernet1/0/0] ospfv3 cost 10
[*DeviceN-GigabitEthernet1/0/0] quit
[*DeviceN] interface gigabitethernet2/0/0
[~DeviceN-GigabitEthernet2/0/0] ospfv3 cost 10
[*DeviceN-GigabitEthernet2/0/0] quit
[*DeviceN] commit
```

**Step 5** Block FRR on the specified interface on Device S.

```
[~DeviceS] interface gigabitethernet1/0/0
[~DeviceS-GigabitEthernet1/0/0] ospfv3 frr block
[*DeviceS-GigabitEthernet1/0/0] quit
[*DeviceS] commit
```

**Step 6** Enable OSPFv3 IP FRR on Device S.

```
[~DeviceS] ospfv3
[*DeviceS-ospfv3-1] frr
[*DeviceS-ospfv3-1-frr] loop-free-alternate
[*DeviceS-ospfv3-1-frr] commit
```

**Step 7** Verify the configuration.

# Run the **display ospfv3 routing** command on Device S to view routing information.

```
[~DeviceS-ospfv3-1-frr] display ospfv3 routing 2001:db8:3000::1 96
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
 N - NSSA
Flags: A - Added to URT6, LT - Locator Routing

OSPFv3 Process (1)
Destination Metric
 Nexthop
2001:DB8:2000:1::/64 3124
 via 2001:DB8:2001::1/96, GE2/0/0
 backup via FE80::2000:10FF:4, GE3/0/0, LFA LINK-NODE
 Priority :Low
```

The preceding command output shows that a backup link has been generated on DeviceS through FRR calculation.

----End

## Configuration Files

- Device S configuration file

```

sysname DeviceS

bfd

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1000::1/96
ospfv3 cost 5

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1001::1/96
ospfv3 cost 15

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1002::1/96
ospfv3 frr block
ospfv3 cost 10

ospfv3 1
router-id 1.1.1.1
bfd all-interfaces enable
bfd all-interfaces frr-binding
frr
loop-free-alternate
area 0.0.0.1

return
```

- Device A configuration file

```

sysname DeviceA

bfd

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1000::2/96
ospfv3 cost 5

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2000::2/96
ospfv3 cost 5

ospfv3 1
router-id 2.2.2.2
bfd all-interfaces enable
bfd all-interfaces frr-binding
frr
loop-free-alternate
area 0.0.0.1

return
```

- Device N configuration file

```

sysname DeviceN
#
```

```
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1002::2/96
ospfv3 cost 10
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2002::2/96
ospfv3 cost 10
#
ospfv3 1
router-id 3.3.3.3
bfd all-interfaces enable
bfd all-interfaces frr-binding
frr
area 0.0.0.1
#
return
```

- Device E configuration file

```
#
sysname DeviceE
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2000::1/96
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2001::1/96
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2002::1/96
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3000::1/96
ospfv3 cost 5
#
ospfv3 1
router-id 4.4.4.4
bfd all-interfaces enable
bfd all-interfaces frr-binding
area 0.0.0.1
#
return
```

## Example for Configuring BFD for OSPFv3

This section describes how to configure BFD for OSPFv3. After BFD for OSPFv3 is configured, BFD can rapidly detect link faults and report them to OSPFv3 so that service traffic can be transmitted through the backup link.

## Networking Requirements

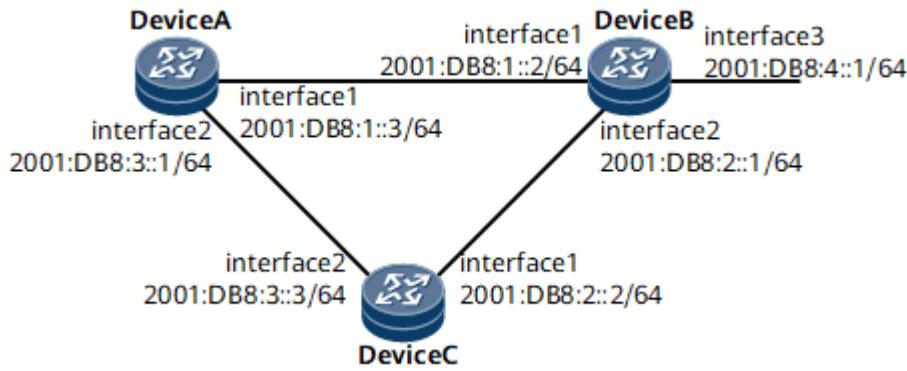
On the network shown in [Figure 1-191](#), the requirements are as follows:

- OSPFv3 runs on Device A, Device B, and Device C.
- BFD is enabled in an OSPFv3 process on Device A, Device B, and Device C.
- Service traffic is transmitted over the primary link (Device A -> Device B). The link (Device A -> Device C -> Device B) functions as the backup link.
- If the link between DeviceA and DeviceB fails, BFD can quickly detect the fault and notify OSPFv3 of the fault so that service traffic can be transmitted through the backup link.

**Figure 1-191** Networking for configuring BFD for OSPFv3

 NOTE

Interfaces 1 through 3 in this example represent GE 1/0/0, GE 1/0/1, and GE 1/0/2, respectively.



## Configuration Notes

To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPFv3 functions on each router.
2. Configure BFD for OSPFv3.

## Data Preparation

To complete the configuration, you need the following data:

- Router ID of Device A: 1.1.1.1
- Router ID of Device B: 2.2.2.2
- Router ID of Device C: 3.3.3.3
- Minimum interval at which BFD packets are received and sent and local detection multiplier on Device A and Device B

## Procedure

**Step 1** Assign an IPv6 address to each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic OSPFv3 functions.

# Configure Device A.

```
[*DeviceA] ospfv3
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet 1/0/1
[*DeviceA-GigabitEthernet1/0/1] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet1/0/1] quit
[*DeviceA] commit
```

# Configure Device B.

```
[*DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] router-id 2.2.2.2
[*DeviceB-ospfv3-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet 1/0/1
[*DeviceB-GigabitEthernet1/0/1] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet1/0/1] quit
[*DeviceB] interface gigabitethernet 1/0/2
[*DeviceB-GigabitEthernet1/0/2] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet1/0/2] commit
```

# Configure Device C.

```
[~DeviceC] ospfv3 1
[~DeviceC-ospfv3-1] router-id 3.3.3.3
[~DeviceC-ospfv3-1] quit
[~DeviceC] interface gigabitethernet 1/0/0
[~DeviceC-GigabitEthernet1/0/0] ospfv3 1 area 0.0.0.0
[~DeviceC-GigabitEthernet1/0/0] quit
[~DeviceC] interface gigabitethernet 1/0/1
[~DeviceC-GigabitEthernet1/0/1] ospfv3 1 area 0.0.0.0
[~DeviceC-GigabitEthernet1/0/1] commit
```

# After the preceding configurations are complete, run the **display ospfv3 peer** command. The command output shows that neighbor relationships are set up. The command output on DeviceA is used as an example.

```
[~DeviceA] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 2
Peer(s) in full state: 2
OSPFv3 Area (0.0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
2.2.2.2 1 Full/DR 00:00:38 GE1/0/0 0
3.3.3.3 1 Full/- 00:00:37 GE1/0/1 0
```

# View information in the OSPFv3 routing table on Device A. The routing table should contain the routes to Device B and Device C.

```
[~DeviceA] display ospfv3 routing
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
 N - NSSA
Flags : A - Added to URT6, LT - Locator Routing
```

| OSPFv3 Process (1)                                                                                                      |        |
|-------------------------------------------------------------------------------------------------------------------------|--------|
| Destination                                                                                                             | Metric |
| 2001:DB8:1::/64<br>directly connected, GE1/0/0, Flags : A                                                               | 1      |
| 2001:DB8:2::/64<br>via FE80::3AE9:7BFF:FE31:307, GE1/0/1, Flags : A<br>via FE80::3AE9:7BFF:FE21:300, GE1/0/0, Flags : A | 2      |
| 2001:DB8:3::/64<br>directly connected, GE1/0/1, Flags : A                                                               | 1      |
| 2001:DB8:4::1/64<br>via FE80::3AE9:7BFF:FE21:300, GE1/0/0                                                               | 1      |

As shown in the OSPFv3 routing table, the next hop address of the route to 2001:DB8:4::1/64 is GigabitEthernet 1/0/0, and traffic is transmitted on the primary link (DeviceA -> DeviceB).

**Step 3** Configure BFD for OSPFv3.

# Enable BFD on Device A globally.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] bfd all-interfaces enable
[*DeviceA-ospfv3-1] bfd all-interfaces min-transmit-interval 100 min-receive-interval 100 detect-multiplier 4
[*DeviceA-ospfv3-1] commit
```

# Enable BFD on Device B globally.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] bfd all-interfaces enable
[*DeviceB-ospfv3-1] bfd all-interfaces min-transmit-interval 100 min-receive-interval 100 detect-multiplier 4
[*DeviceB-ospfv3-1] commit
```

# Enable BFD on Device C globally.

```
[~DeviceC] bfd
[*DeviceC-bfd] quit
[*DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] bfd all-interfaces enable
[*DeviceC-ospfv3-1] bfd all-interfaces min-transmit-interval 100 min-receive-interval 100 detect-multiplier 4
[*DeviceC-ospfv3-1] commit
```

# After the preceding configurations are complete, run the **display ospfv3 bfd session** command on Device A or Device B. You can view that the status of the BFD session is Up. Use the command output on DeviceB as an example.

```
[~DeviceB] display ospfv3 bfd session
* - STALE
```

OSPFv3 Process (1)

| Neighbor-Id | Interface | BFD-Status |
|-------------|-----------|------------|
| 1.1.1.1     | GE1/0/0   | Up         |
| 3.3.3.3     | GE1/0/1   | Up         |

**Step 4** Verify the configuration.

# Run the **shutdown** command on GE 1/0/0 of Device B to simulate a primary link failure.

```
[~DeviceB] interface gigabitethernet1/0/0
```

```
[~DeviceB-GigabitEthernet1/0/0] shutdown
[*DeviceB-GigabitEthernet1/0/0] commit
```

# Check the routing table on routerDevice A. The command output shows that the next hop of the route to 2001:DB8:4::1/64 is changed to GigabitEthernet 1/0/1. Therefore, traffic is switched to the backup link (DeviceA -> DeviceC -> DeviceB).

```
[~DeviceA] display ospfv3 routing
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
N - NSSA
Flags : A - Added to URT6, LT - Locator Routing

OSPFv3 Process (1)
Destination Metric
Next-hop
2001:DB8:2::/64 2
via FE80::3AE9:7BFF:FE31:307, GE1/0/1, Flags : A
2001:DB8:3::/64 1
directly connected, GE1/0/1, Flags : A
2001:DB8:4::/64 3
via FE80::3AE9:7BFF:FE31:307, GE1/0/1, Flags : A
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

bfd

ospfv3 1
router-id 1.1.1.1
bfd all-interfaces enable
bfd all-interfaces min-transmit-interval 100 min-receive-interval 100 detect-multiplier 4
area 0.0.0.0

interface gigabitethernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::3/64
ospfv3 1 area 0.0.0.0

interface gigabitethernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
ospfv3 1 area 0.0.0.0

return
```

- Device B configuration file

```

sysname DeviceB

bfd

ospfv3 1
router-id 2.2.2.2
bfd all-interfaces enable
bfd all-interfaces min-transmit-interval 100 min-receive-interval 100 detect-multiplier 4
area 0.0.0.0

interface gigabitethernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
```

```
ospfv3 1 area 0.0.0.0
#
interface gigabitethernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
ospfv3 1 area 0.0.0.0
#
interface gigabitethernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
ospfv3 1 area 0.0.0.0
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
bfd
#
ospfv3 1
router-id 3.3.3.3
bfd all-interfaces enable
bfd all-interfaces min-transmit-interval 100 min-receive-interval 100 detect-multiplier 4
area 0.0.0.0
#
interface gigabitethernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
ospfv3 1 area 0.0.0.0
#
interface gigabitethernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::3/64
ospfv3 1 area 0.0.0.0
#
return
```

## Example for Configuring IPsec for OSPFv3

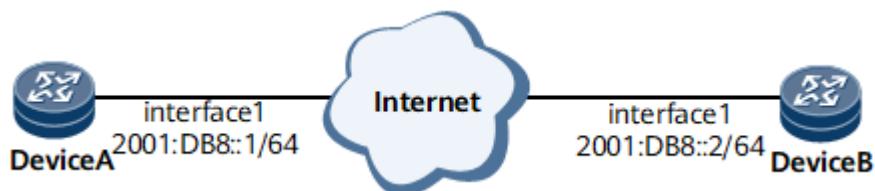
### Networking Requirements

On the network shown in [Figure 1-192](#), DeviceA and DeviceB are connected over a public network, and OSPFv3 is deployed for their communication.

**Figure 1-192** Configuring IPsec



Interface1 in this example represents GE 1/0/1.



If no authentication mechanism is configured, routing protocol packets transmitted between DeviceA and DeviceB may be modified or spoofed by

attackers. As a result, their connection may be torn down, or incorrect routes may be imported.

To prevent such attacks, establish an IPsec tunnel between DeviceA and DeviceB to protect OSPFv3 packets transmitted between them. Configure Encapsulating Security Payload (ESP) as the security protocol, and Secure Hash Algorithm 2-256 (SHA2-256) as the authentication algorithm.

## Configuration Notes

- The encapsulation modes and security protocols on both IPsec peers must be identical.
- The authentication modes and encryption algorithms on the two IPsec peers must be identical.
- The security parameter indexes (SPIs) and authentication keys on the two IPsec peers must be identical.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPFv3 functions on DeviceA and DeviceB.
2. Configure IPsec proposals. Configure ESP as the security protocol, SHA2-256 as the authentication algorithm, and AES-256 as the encryption algorithm.
3. Set SA parameters.
4. Apply SAs to OSPFv3 processes, enabling IPsec to protect OSPFv3 packets transmitted between DeviceA and DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

| Device Name | Router ID | Process ID | SPI   | Authentication Key in String Format |
|-------------|-----------|------------|-------|-------------------------------------|
| DeviceA     | 1.1.1.1   | 1          | 12345 | abcdef                              |
| DeviceB     | 2.2.2.2   | 1          | 12345 | abcdef                              |

## Procedure

### Step 1 Configure OSPFv3 on DeviceA and DeviceB.

# Configure DeviceA.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
[*HUAWEI] commit
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] area 0
[*DeviceA-ospfv3-1-area-0.0.0.0] commit
```

```
[~DeviceA-ospfv3-1-area-0.0.0] quit
```

```
Configure DeviceB.
```

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] router-id 2.2.2.2
[*DeviceB-ospfv3-1] area 0
[*DeviceB-ospfv3-1-area-0.0.0] commit
[~DeviceB-ospfv3-1-area-0.0.0] quit
```

**Step 2** Configure IPv6 addresses for and enable OSPFv3 on interfaces.

```
Configure DeviceA.
```

```
[~DeviceA] interface gigabitethernet1/0/1
[~DeviceA-GigabitEthernet1/0/1] ipv6 enable
[*DeviceA-GigabitEthernet1/0/1] ipv6 address 2001:db8::1 64
[*DeviceA-GigabitEthernet1/0/1] ospfv3 1 area 0
[*DeviceA-GigabitEthernet1/0/1] commit
[~DeviceA-GigabitEthernet1/0/1] quit
```

```
Configure DeviceB.
```

```
[~DeviceB] interface gigabitethernet1/0/1
[~DeviceB-GigabitEthernet1/0/1] ipv6 enable
[*DeviceB-GigabitEthernet1/0/1] ipv6 address 2001:db8::2 64
[*DeviceB-GigabitEthernet1/0/1] ospfv3 1 area 0
[*DeviceB-GigabitEthernet1/0/1] commit
[~DeviceB-GigabitEthernet1/0/1] quit
```

**Step 3** Configure security proposals on DeviceA and DeviceB.

```
Configure a security proposal on DeviceA.
```

```
[~DeviceA] ipsec proposal proposal1
[*DeviceA-ipsec-proposal-proposal1] encapsulation-mode transport
[*DeviceA-ipsec-proposal-proposal1] transform esp
[*DeviceA-ipsec-proposal-proposal1] esp encryption-algorithm aes 256
[*DeviceA-ipsec-proposal-proposal1] esp authentication-algorithm sha2-256
[*DeviceA-ipsec-proposal-proposal1] commit
[~DeviceA-ipsec-proposal-proposal1] quit
```

```
Configure a security proposal on DeviceB.
```

```
[~DeviceB] ipsec proposal proposal2
[*DeviceB-ipsec-proposal-proposal2] encapsulation-mode transport
[*DeviceB-ipsec-proposal-proposal2] transform esp
[*DeviceB-ipsec-proposal-proposal2] esp encryption-algorithm aes 256
[*DeviceB-ipsec-proposal-proposal2] esp authentication-algorithm sha2-256
[*DeviceB-ipsec-proposal-proposal2] commit
[~DeviceB-ipsec-proposal-proposal2] quit
```

```
Run the display ipsec proposal command on DeviceA and DeviceB to check the configuration. The following example uses the command output on DeviceA.
```

```
[~DeviceA] display ipsec proposal
Total IPsec proposal number: 1
IPsec proposal name: proposal1
encapsulation mode: transport
transform: esp-new
ESP protocol: authentication SHA2-HMAC-256, encryption 256-AES
```

**Step 4** Configure IPsec SAs on DeviceA and DeviceB and apply a proposal to each SA.

```
Configure an IPsec SA on DeviceA and apply a proposal to it.
```

```
[~DeviceA] ipsec sa sa1
```

```
[*DeviceA-ipsec-sa-sa1] proposal proposal1
[*DeviceA-ipsec-sa-sa1] commit
```

# Configure an IPsec SA on DeviceB and apply a proposal to it.

```
[~DeviceB] ipsec sa sa2
[*DeviceB-ipsec-sa-sa2] proposal proposal2
[*DeviceB-ipsec-sa-sa2] commit
```

**Step 5** # Configure SPIs and authentication keys in string format on DeviceA and DeviceB.

# Configure SPIs and authentication keys in string format on DeviceA.

```
[~DeviceA] ipsec sa sa1
[*DeviceA-ipsec-sa-sa1] sa spi inbound esp 12345
[*DeviceA-ipsec-sa-sa1] sa spi outbound esp 12345
[*DeviceA-ipsec-sa-sa1] sa string-key inbound esp abcdef
[*DeviceA-ipsec-sa-sa1] sa string-key outbound esp abcdef
[*DeviceA-ipsec-sa-sa1] commit
[~DeviceA-ipsec-sa-sa1] quit
```

# Configure SPIs and authentication keys in string format on DeviceB.

```
[~DeviceB] ipsec sa sa2
[*DeviceB-ipsec-sa-sa2] sa spi outbound esp 12345
[*DeviceB-ipsec-sa-sa2] sa spi inbound esp 12345
[*DeviceB-ipsec-sa-sa2] sa string-key outbound esp abcdef
[*DeviceB-ipsec-sa-sa2] sa string-key inbound esp abcdef
[*DeviceB-ipsec-sa-sa2] commit
[~DeviceB-ipsec-sa-sa2] quit
```

**Step 6** Apply SAs to OSPFv3 processes.

# Apply an SA to an OSPFv3 process on DeviceA.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] ipsec sa sa1
[*DeviceA-ospfv3-1] commit
```

# Apply an SA to an OSPFv3 process on DeviceB.

```
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] ipsec sa sa2
[*DeviceB-ospfv3-1] commit
```

**Step 7** Verify the configuration.

# Run the **display ipsec sa** command on DeviceA and DeviceB to check the configuration. The following example uses the command output on DeviceA.

```
[~DeviceA] display ipsec sa
Total IP security association number: 1
IP security association name: sa1
Number of references: 1
proposal name: proposal1
State: Complete
inbound AH setting:
 AH spi:
 AH string-key:
 AH authentication hex key:
inbound ESP setting:
 ESP spi: 12345 (0x3039)
 ESP string-key: %##%#<}jb{br9\zi%X+/Y@:Y>Lw(L\v#*^KsM"/8RaRe$%#%#
 ESP encryption hex key:
 ESP authentication hex key:
outbound AH setting:
 AH spi:
 AH string-key:
 AH authentication hex key:
```

```
outbound ESP setting:
ESP spi: 12345 (0x3039)
ESP string-key: %##%#<]j/@X4355SE9JZTD0>GQf"}w2@X,k6.E\Z,z\{##%#%#
ESP encryption hex key:
ESP authentication hex key:
```

# Run the **display ipsec statistics** command to check statistics about incoming and outgoing protocol packets processed by IPsec and detailed information about dropped protocol packets. If statistics about incoming and outgoing protocol packets processed by IPsec are displayed, the configuration succeeds. For example:

```
[~DeviceA] display ipsec statistics
IPv6 security packet statistics:
 input/output security packets: 184/19
 input/output security bytes: 13216/1312
 input/output dropped security packets: 0/0
 dropped security packet detail:
 memory process problem: 0
 can't find SA: 0
 queue is full: 0
 authentication is failed: 0
 wrong length: 0
 replay packet: 0
 too long packet: 0
 invalid SA: 0
 policy deny: 0
the normal packet statistics:
 input/output dropped normal packets: 0/0
IPv4 security packet statistics:
 input/output security packets: 0/0
 input/output security bytes: 0/0
 input/output dropped security packets: 0/0
 dropped security packet detail:
 memory process problem: 0
 can't find SA: 0
 queue is full: 0
 authentication is failed: 0
 wrong length: 0
 replay packet: 0
 too long packet: 0
 invalid SA: 0
 policy deny: 0
the normal packet statistics:
 input/output dropped normal packets: 0/0
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

ipsec proposal proposal1
encapsulation-mode transport
esp authentication-algorithm sha2-256
esp encryption-algorithm aes 256

ipsec sa sa1
proposal proposal1
sa spi inbound esp 12345
sa string-key inbound esp %##%#<]jb{br9\zi%X+/:Y>Lw(L\v#^KsM"/8RaRe$%#%#
sa spi outbound esp 12345
sa string-key outbound esp %##%#<]j/@X4355SE9JZTD0>GQf"}w2@X,k6.E\Z,z\{##%#%#

ospfv3 1
router-id 1.1.1.1
```

```
ipsec sa sa1
area 0.0.0.
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8::1/64
ospfv3 1 area 0
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
ipsec proposal proposal2
encapsulation-mode transport
esp authentication-algorithm sha2-256
esp encryption-algorithm aes 256
#
ipsec sa sa2
proposal proposal2
sa spi inbound esp 12345
sa string-key inbound esp %##<}@XSE9JZT5]2"t#]2"t<}@XSE9JZT5>%##%
sa spi outbound esp 12345
sa string-key outbound esp %##%)YTP%@nFE7bL^B&WSBiQ1[p#M"/8RaRe%7%#%
#
ospfv3 1
router-id 2.2.2.2
ipsec sa sa2
area 0.0.0.
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8::2/64
ospfv3 1 area 0
#
return
```

## Example for Configuring OSPFv3 Route Summarization on an ASBR

This section provides an example showing how to configure OSPFv3 route summarization on an ASBR.

### Networking Requirements

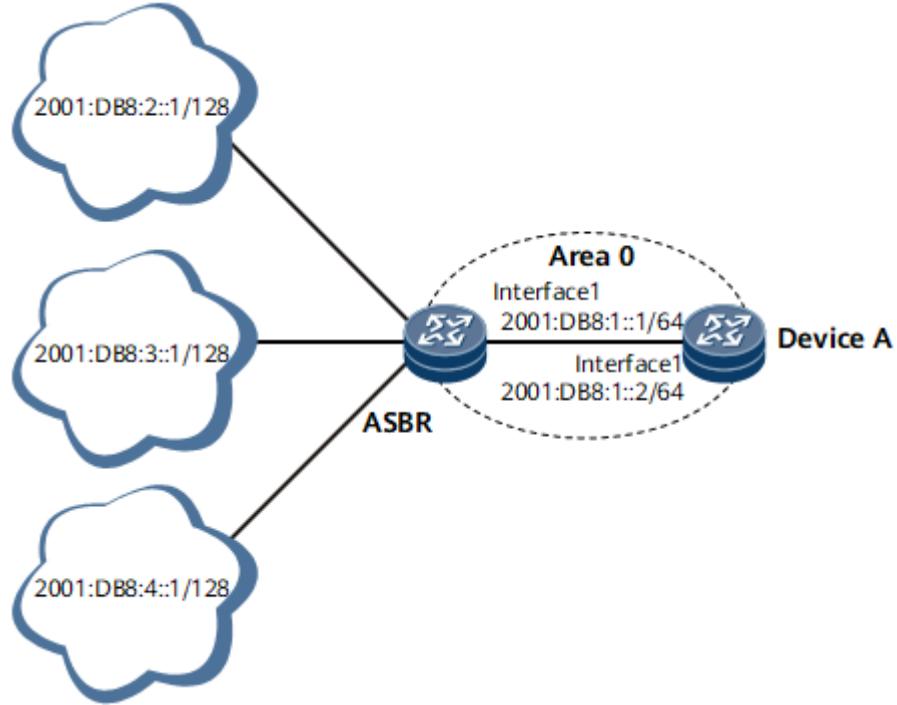
Routes with the same IPv6 prefix can be summarized into one route. On a large-scale OSPFv3 network, route lookup may slow down because of the large size of the routing table. To reduce the routing table size and simplify management, configure route summarization. With route summarization, if a link connected to a device within the IPv6 address range that has been summarized alternates between up and down, the link status change is not advertised to the devices beyond the IPv6 address range. This prevents route flapping and improves network stability.

In [Figure 1-193](#), both the ASBR and Device A run OSPFv3. The ASBR imports three static routes with the same prefix: 2001:DB8:2::1/128, 2001:DB8:3::1/128, and 2001:DB8:4::1/128. After the three static routes with the same prefix are summarized into route 2001:DB8::/32 on the ASBR, the ASBR advertises only this route to area 0. This reduces the size of the routing table, simplifies route management, and improves network stability.

**Figure 1-193** Example for configuring OSPFv3 route summarization on an ASBR

 **NOTE**

Interface 1 in this example represents GE 1/0/0.



## Configuration Notes

To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

## Configuration Roadmap

The configuration roadmap is as follows:

1. Assign an IP address to each interface to ensure that devices on the network can communicate with each other.
2. Configure basic OSPFv3 functions on the ASBR and Device A.
3. Configure three static routes (2001:DB8:2::1/128, 2001:DB8:3::1/128, and 2001:DB8:4::1/128) on the ASBR and import them into OSPFv3.
4. Configure OSPFv3 route summarization on the ASBR.

## Data Preparation

To complete the configuration, you need the following data:

- Area 0
- Router ID (1.1.1.1) of Device A
- Router ID (2.2.2.2) of the ASBR

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see "Configuration Files" in this section.

**Step 2** Configure basic OSPFv3 functions.

# Configure Device A.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] area 0.0.0.0
[*DeviceA-ospfv3-1-area-0.0.0.0] quit
[*DeviceA-ospfv3-1] quit
[*DeviceA] commit
[~DeviceA] interface gigabitethernet1/0/0
[~DeviceA-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] commit
```

# Configure the ASBR.

```
[~ASBR] ospfv3 1
[*ASBR-ospfv3-1] router-id 2.2.2.2
[*ASBR-ospfv3-1] area 0.0.0.0
[*ASBR-ospfv3-1-area-0.0.0.0] quit
[*ASBR-ospfv3-1] quit
[*ASBR] commit
[~ASBR] interface gigabitethernet1/0/0
[~ASBR-GigabitEthernet1/0/0] ospfv3 1 area 0
[*ASBR-GigabitEthernet1/0/0] quit
[*ASBR] commit
```

# After the configuration is complete, run the **display ospfv3 peer** command. The command output shows that a neighbor relationship has been established between DeviceA and the ASBR. The following example uses the command output on the ASBR:

```
[~ASBR] display ospfv3 peer

OSPFv3 Process (1)
Total number of peer(s): 1
Peer(s) in full state: 1
OSPFv3 Area (0.0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
1.1.1.1 1 Full/DR 00:00:37 GE1/0/0 0
```

**Step 3** Configure three static routes with the same prefix on the ASBR and import them into OSPFv3.

```
[~ASBR] ipv6 route-static 2001:DB8:2::1 128 NULL0
[*ASBR] ipv6 route-static 2001:DB8:3::1 128 NULL0
[*ASBR] ipv6 route-static 2001:DB8:4::1 128 NULL0
[*ASBR] commit
[~ASBR] ospfv3 1
[*ASBR-ospfv3-1] import-route static
[*ASBR-ospfv3-1] quit
[*ASBR] commit
```

# After the configuration is complete, run the **display ospfv3 lsdb** command on the ASBR to check OSPFv3 LSDB information. There are three AS-external LSAs in the LSDB, and the prefixes of corresponding routes are 2001:DB8:2::/128, 2001:DB8:3::/128, and 2001:DB8:4::/128, indicating that route summarization is not performed.

```
[~ASBR] display ospfv3 lsdb
OSPFv3 Router with ID (2.2.2.2) (Process 1)
```

```
Link-LSA (Interface GE1/0/0)
Link State ID Origin Router Age Seq# CkSum Prefix
0.0.0.18 1.1.1.1 172 0x80000002 0xa78d 1
0.0.0.18 2.2.2.2 117 0x80000002 0xde3b 1

Router-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum Link
0.0.0.1 1.1.1.1 115 0x80000002 0xd020 1
0.0.0.1 2.2.2.2 80 0x80000003 0xb633 1

Network-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum
0.0.0.18 1.1.1.1 115 0x80000001 0xd333

Intra-Area-Prefix-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum Prefix Reference
0.0.0.1 1.1.1.1 115 0x80000001 0x279b 1 Network-LSA

AS-External-LSA
Link State ID Origin Router Age Seq# CkSum Type
0.0.0.1 2.2.2.2 76 0x80000001 0xade0 E2
0.0.0.2 2.2.2.2 76 0x80000001 0xb3d8 E2
0.0.0.3 2.2.2.2 76 0x80000001 0xb9d0 E2
[~ASBR] display ospfv3 lsdb external

OSPFV3 Router with ID (2.2.2.2) (Process 1)

AS-External-LSA

LS Age: 116
LS Type: AS-External-LSA
Link State ID: 0.0.0.1
Originating Router: 2.2.2.2
LS Seq Number: 0x80000001
Retransmit Count: 0
Checksum: 0xade0
Length: 48
Flags: (E|-|T)
Metric Type: 2 (Larger than any link state path)
Metric: 1
Prefix: 2001:DB8:2::1/128
Prefix Options: 0 (-|-|-|-)
Tag: 1

LS Age: 116
LS Type: AS-External-LSA
Link State ID: 0.0.0.2
Originating Router: 2.2.2.2
LS Seq Number: 0x80000001
Retransmit Count: 0
Checksum: 0xb3d8
Length: 48
Flags: (E|-|T)
Metric Type: 2 (Larger than any link state path)
Metric: 1
Prefix: 2001:DB8:3::1/128
Prefix Options: 0 (-|-|-|-)
Tag: 1

LS Age: 116
LS Type: AS-External-LSA
Link State ID: 0.0.0.3
Originating Router: 2.2.2.2
LS Seq Number: 0x80000001
Retransmit Count: 0
Checksum: 0xb9d0
Length: 48
Flags: (E|-|T)
```

```
Metric Type: 2 (Larger than any link state path)
Metric: 1
Prefix: 2001:DB8:4::1/128
Prefix Options: 0 (-|-|-|-)
Tag: 1
```

**Step 4** Configure route summarization on the ASBR.

```
On the ASBR, summarize three static routes 2001:DB8:2::1/128,
2001:DB8:3::1/128, and 2001:DB8:4::1/128 into route 2001:DB8::/32.
```

```
[~ASBR] ospfv3 1
[*ASBR-ospfv3-1] asbr-summary 2001:DB8:: 32
[*ASBR-ospfv3-1] quit
[*ASBR] commit
```

**Step 5** Verify the configuration.

```
After route summarization is configured, run the display ospfv3 lsdb command
on the ASBR. The command output shows that the three static routes
2001:DB8:2::1/128, 2001:DB8:3::1/128, and 2001:DB8:4::1/128 have been
summarized into route 2001:DB8::/32.
```

```
[~ASBR] display ospfv3 lsdb
OSPFv3 Router with ID (2.2.2.2) (Process 1)

Link-LSA (Interface GE1/0/0)
Link State ID Origin Router Age Seq# CkSum Prefix
0.0.0.18 1.1.1.1 643 0x80000002 0xa78d 1
0.0.0.18 2.2.2.2 588 0x80000002 0xde3b 1

Router-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum Link
0.0.0.1 1.1.1.1 586 0x80000002 0xd020 1
0.0.0.1 2.2.2.2 551 0x80000003 0xb633 1

Network-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum
0.0.0.18 1.1.1.1 586 0x80000001 0xd333

Intra-Area-Prefix-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum Prefix Reference
0.0.0.1 1.1.1.1 586 0x80000001 0x279b 1 Network-LSA

AS-External-LSA
Link State ID Origin Router Age Seq# CkSum Type
0.0.0.4 2.2.2.2 123 0x80000001 0x606f E2
[~ASBR] display ospfv3 lsdb external

OSPFv3 Router with ID (2.2.2.2) (Process 1)

AS-External-LSA

LS Age: 133
LS Type: AS-External-LSA
Link State ID: 0.0.0.4
Originating Router: 2.2.2.2
LS Seq Number: 0x80000001
Retransmit Count: 0
Checksum: 0x606f
Length: 36
Flags: (E|-|T)
Metric Type: 2 (Larger than any link state path)
Metric: 2
Prefix: 2001:DB8::/32
Prefix Options: 0 (-|-|-|-)
Tag: 1
```

# Run the **display ospfv3 asbr-summary** command on the ASBR. The command output shows information about the summary route of the static routes imported by OSPFv3.

```
[~ASBR] display ospfv3 asbr-summary

OSPFv3 Process (1)
Prefix Prefix-Len Matched Status
2001:DB8:: 32 3 [Active] Advertised

----End
```

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
ospfv3 1
router-id 1.1.1.1
area 0.0.0.0
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:DB8:1::2/64
ospfv3 1 area 0.0.0.0
#
return
```

- ASBR configuration file

```
#
sysname ASBR
#
ospfv3 1
router-id 2.2.2.2
import-route static
asbr-summary 2001:DB8:: 16
area 0.0.0.0
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:DB8:1::1/64
ospfv3 1 area 0.0.0.0
#
ipv6 route-static 2001:DB8:2::1 128 NULL0
ipv6 route-static 2001:DB8:3::1 128 NULL0
ipv6 route-static 2001:DB8:4::1 128 NULL0
#
return
```

## Example for Configuring OSPFv3 Route Summarization on an ABR

This section provides an example showing how to configure OSPFv3 route summarization on an ABR.

## Networking Requirements

Routes with the same IPv6 prefix can be summarized into one route. On a large-scale OSPFv3 network, route lookup may slow down because of the large size of the routing table. To reduce the routing table size and simplify management, configure route summarization. With route summarization, if a link connected to a device within an IPv6 address range that has been summarized alternates

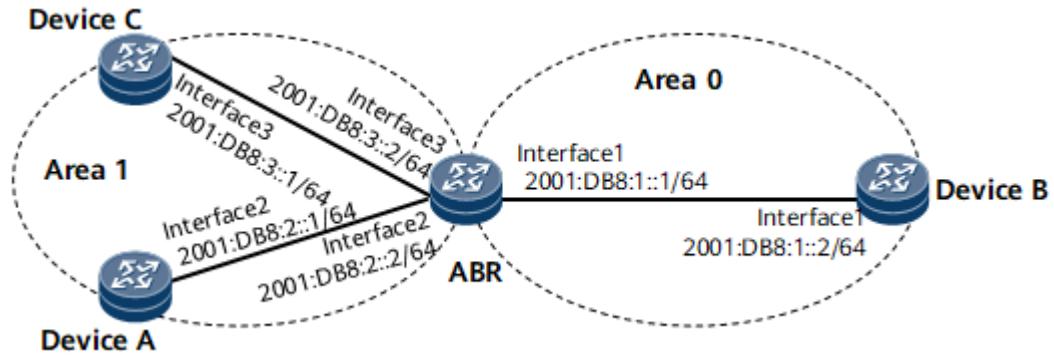
between up and down, the link status change is not advertised to the devices beyond the IPv6 address range. This prevents route flapping and improves network stability.

In [Figure 1-194](#), all devices run OSPFv3. To reduce the routing table size, simplify route management, and improve network stability, it is required that the ABR be configured to summarize area 1's routes with the same prefix (2001:DB8::) into route 2001:DB8::/32 and advertise only the summary route to area 0.

**Figure 1-194** Example for configuring OSPFv3 route summarization on an ABR

 NOTE

Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



## Configuration Notes

To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

## Configuration Roadmap

The configuration roadmap is as follows:

1. Assign an IP address to each interface to ensure that devices on the network can communicate with each other.
2. Configure basic OSPFv3 functions on all devices.
3. Configure OSPFv3 route summarization on the ABR.

## Data Preparation

To complete the configuration, you need the following data:

- Areas 0 and 1
- Router ID (1.1.1.1) of Device B
- Router ID (2.2.2.2) of the ABR
- Router ID (3.3.3.3) of Device A

- Router ID (4.4.4.4) of Device C

## Procedure

- Step 1** Configure an IP address for each interface. For configuration details, see "Configuration Files" in this section.
- Step 2** Configure basic OSPFv3 functions.

# Configure Device A.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] router-id 3.3.3.3
[*DeviceA-ospfv3-1] area 0.0.0.1
[*DeviceA-ospfv3-1-area-0.0.0.1] quit
[*DeviceA-ospfv3-1] quit
[*DeviceA] commit
[~DeviceA] interface gigabitethernet2/0/0
[~DeviceA-GigabitEthernet2/0/0] ospfv3 1 area 1
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

# Configure Device B.

```
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] router-id 1.1.1.1
[*DeviceB-ospfv3-1] area 0.0.0.0
[*DeviceB-ospfv3-1-area-0.0.0.0] quit
[*DeviceB-ospfv3-1] quit
[*DeviceB] commit
[~DeviceB] interface gigabitethernet1/0/0
[~DeviceB-GigabitEthernet1/0/0] ospfv3 1 area 0
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] commit
```

# Configure Device C.

```
[~DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] router-id 4.4.4.4
[*DeviceC-ospfv3-1] area 0.0.0.1
[*DeviceC-ospfv3-1-area-0.0.0.1] quit
[*DeviceC-ospfv3-1] quit
[*DeviceC] commit
[~DeviceC] interface gigabitethernet3/0/0
[~DeviceC-GigabitEthernet3/0/0] ospfv3 1 area 1
[*DeviceC-GigabitEthernet3/0/0] quit
[*DeviceC] commit
```

# Configure the ABR.

```
[~ABR] ospfv3 1
[*ABR-ospfv3-1] router-id 2.2.2.2
[*ABR-ospfv3-1] area 0.0.0.0
[*ABR-ospfv3-1-area-0.0.0.0] quit
[*ABR-ospfv3-1] area 0.0.0.1
[*ABR-ospfv3-1-area-0.0.0.1] quit
[*ABR-ospfv3-1] quit
[*ABR] commit
[~ABR] interface gigabitethernet1/0/0
[~ABR-GigabitEthernet1/0/0] ospfv3 1 area 0
[*ABR-GigabitEthernet1/0/0] quit
[*ABR] interface gigabitethernet2/0/0
[~ABR-GigabitEthernet2/0/0] ospfv3 1 area 1
[*ABR-GigabitEthernet2/0/0] quit
[*ABR] interface gigabitethernet3/0/0
[~ABR-GigabitEthernet3/0/0] ospfv3 1 area 1
[*ABR-GigabitEthernet3/0/0] quit
[*ABR] commit
```

# Run the **display ospfv3 peer** command to check whether the ABR establishes a neighbor relationship with Device A, Device B, and Device C. The following example uses the command output on the ABR:

```
[~ABR] display ospfv3 peer

OSPFv3 Process (1)
Total number of peer(s): 3
Peer(s) in full state: 3
OSPFv3 Area (0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
1.1.1.1 1 Full/DR 00:00:35 GE1/0/0 0
OSPFv3 Area (0.0.1)
Neighbor ID Pri State Dead Time Interface Instance ID
4.4.4.4 1 Full/Backup 00:00:40 GE3/0/0 0
3.3.3.3 1 Full/Backup 00:00:31 GE2/0/0 0
```

# Run the **display ospfv3 lsdb** command on the ABR to check OSPFv3 LSDB information. The **Inter-area-prefix LSA** field in the LSDB of area 1 shows that no summarization is performed for the routes. Therefore, the routes advertised to area 0 are not summarized.

```
[~ABR] display ospfv3 lsdb

OSPFv3 Router with ID (2.2.2.2) (Process 1)

Link-LSA (Interface GE1/0/0)
Link State ID Origin Router Age Seq# CkSum Prefix
0.0.0.18 1.1.1.1 1740 0x80000010 0xb9b 1
0.0.0.18 2.2.2.2 1684 0x80000010 0xc249 1

Link-LSA (Interface GE3/0/0)
Link State ID Origin Router Age Seq# CkSum Prefix
0.0.0.19 2.2.2.2 154 0x80000002 0x33e2 1
0.0.0.19 4.4.4.4 75 0x80000002 0xf927 1

Link-LSA (Interface GE2/0/0)
Link State ID Origin Router Age Seq# CkSum Prefix
0.0.0.20 2.2.2.2 383 0x8000000e 0x306 1
0.0.0.11 3.3.3.3 232 0x8000000e 0xc146 1

Router-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum Link
0.0.0.1 1.1.1.1 1682 0x80000010 0xb42e 1
0.0.0.1 2.2.2.2 384 0x80000014 0x9148 1

Network-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum
0.0.0.18 1.1.1.1 1682 0x8000000f 0xb741

Inter-Area-Prefix-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum
0.0.0.2 2.2.2.2 154 0x80000001 0x9071
0.0.0.3 2.2.2.2 1726 0x8000000a 0xf315

Intra-Area-Prefix-LSA (Area 0.0.0.0)
Link State ID Origin Router Age Seq# CkSum Prefix Reference
0.0.0.1 1.1.1.1 1682 0x8000000f 0xba9 1 Network-LSA

Router-LSA (Area 0.0.0.1)
Link State ID Origin Router Age Seq# CkSum Link
0.0.0.1 2.2.2.2 70 0x8000000f 0xa0f4 2
0.0.0.1 3.3.3.3 226 0x8000000e 0x904d 1
0.0.0.1 4.4.4.4 71 0x80000002 0xe8f5 1

Network-LSA (Area 0.0.0.1)
Link State ID Origin Router Age Seq# CkSum
0.0.0.19 2.2.2.2 70 0x80000001 0x22d3
```

```
0.0.0.20 2.2.2.2 225 0x8000000d 0xdd0f
 Inter-Area-Prefix-LSA (Area 0.0.0.1)
Link State ID Origin Router Age Seq# CkSum
0.0.0.1 2.2.2.2 390 0x8000000d 0x6296

 Intra-Area-Prefix-LSA (Area 0.0.0.1)
Link State ID Origin Router Age Seq# CkSum Prefix Reference
0.0.0.1 2.2.2.2 231 0x8000000d 0x4f5c 1 Network-LSA
0.0.0.2 2.2.2.2 76 0x80000001 0x6b4b 1 Network-LSA
[~ABR] display ospfv3 lsdb inter-prefix

OSPFv3 Router with ID (2.2.2.2) (Process 1)

Inter-Area-Prefix-LSA (Area 0.0.0.0)

LS Age: 197
LS Type: Inter-Area-Prefix-LSA
Link State ID: 0.0.0.2
Originating Router: 2.2.2.2
LS Seq Number: 0x80000001
Retransmit Count: 0
Checksum: 0x9071
Length: 36

Metric: 1
Prefix: 2001:DB8:3::/64
Prefix Options: 0 (-|-|-|-|)

LS Age: 1769
LS Type: Inter-Area-Prefix-LSA
Link State ID: 0.0.0.3
Originating Router: 2.2.2.2
LS Seq Number: 0x8000000a
Retransmit Count: 0
Checksum: 0xf315
Length: 36

Metric: 1
Prefix: 2001:DB8:2::/48
Prefix Options: 0 (-|-|-|-|)

Inter-Area-Prefix-LSA (Area 0.0.0.1)

LS Age: 427
LS Type: Inter-Area-Prefix-LSA
Link State ID: 0.0.0.1
Originating Router: 2.2.2.2
LS Seq Number: 0x8000000d
Retransmit Count: 0
Checksum: 0x6296
Length: 36

Metric: 1
Prefix: 2001:DB8:1::/64
Prefix Options: 0 (-|-|-|-|)
```

**Step 3** Configure the ABR to summarize the routes with the same prefix in area 1 into route 2001:DB8::/32.

```
[~ABR] ospfv3 1
[*ABR-ospfv3-1] area 0.0.0.1
[*ABR-ospfv3-1-area-0.0.0.1] abr-summary 2001:DB8:: 32
[*ABR-ospfv3-1-area-0.0.0.1] quit
[*ABR-ospfv3-1] quit
[*ABR] commit
```

**Step 4** Verify the configuration.

# After route summarization is configured, run the **display ospfv3 lsdb** command on the ABR. The command output shows that the routes with the same prefix in area 1 are summarized into route 2001:DB8::/32 and advertised to area 0.

[~ABR] **display ospfv3 lsdb**

OSPFv3 Router with ID (2.2.2.2) (Process 1)

Link-LSA (Interface GE1/0/0)

| Link State ID | Origin Router | Age | Seq#       | CkSum  | Prefix |
|---------------|---------------|-----|------------|--------|--------|
| 0.0.0.18      | 1.1.1.1       | 108 | 0x80000011 | 0x899c | 1      |
| 0.0.0.18      | 2.2.2.2       | 52  | 0x80000011 | 0xc04a | 1      |

Link-LSA (Interface GE3/0/0)

| Link State ID | Origin Router | Age | Seq#       | CkSum  | Prefix |
|---------------|---------------|-----|------------|--------|--------|
| 0.0.0.19      | 2.2.2.2       | 322 | 0x80000002 | 0x33e2 | 1      |
| 0.0.0.19      | 4.4.4.4       | 243 | 0x80000002 | 0xf927 | 1      |

Link-LSA (Interface GE2/0/0)

| Link State ID | Origin Router | Age | Seq#       | CkSum  | Prefix |
|---------------|---------------|-----|------------|--------|--------|
| 0.0.0.20      | 2.2.2.2       | 551 | 0x8000000e | 0x306  | 1      |
| 0.0.0.11      | 3.3.3.3       | 400 | 0x8000000e | 0xc146 | 1      |

Router-LSA (Area 0.0.0.0)

| Link State ID | Origin Router | Age | Seq#       | CkSum  | Link |
|---------------|---------------|-----|------------|--------|------|
| 0.0.0.1       | 1.1.1.1       | 50  | 0x80000011 | 0xb22f | 1    |
| 0.0.0.1       | 2.2.2.2       | 552 | 0x80000014 | 0x9148 | 1    |

Network-LSA (Area 0.0.0.0)

| Link State ID | Origin Router | Age | Seq#       | CkSum  |
|---------------|---------------|-----|------------|--------|
| 0.0.0.18      | 1.1.1.1       | 50  | 0x80000010 | 0xb542 |

Inter-Area-Prefix-LSA (Area 0.0.0.0)

| Link State ID | Origin Router | Age | Seq#       | CkSum  |
|---------------|---------------|-----|------------|--------|
| 0.0.0.3       | 2.2.2.2       | 30  | 0x80000001 | 0x6dba |

Intra-Area-Prefix-LSA (Area 0.0.0.0)

| Link State ID | Origin Router | Age | Seq#       | CkSum | Prefix Reference |
|---------------|---------------|-----|------------|-------|------------------|
| 0.0.0.1       | 1.1.1.1       | 50  | 0x80000010 | 0x9aa | 1 Network-LSA    |

Router-LSA (Area 0.0.0.1)

| Link State ID | Origin Router | Age | Seq#       | CkSum  | Link |
|---------------|---------------|-----|------------|--------|------|
| 0.0.0.1       | 2.2.2.2       | 238 | 0x8000000f | 0xa0f4 | 2    |
| 0.0.0.1       | 3.3.3.3       | 394 | 0x8000000e | 0x904d | 1    |
| 0.0.0.1       | 4.4.4.4       | 239 | 0x80000002 | 0xe8f5 | 1    |

Network-LSA (Area 0.0.0.1)

| Link State ID | Origin Router | Age | Seq#       | CkSum  |
|---------------|---------------|-----|------------|--------|
| 0.0.0.19      | 2.2.2.2       | 238 | 0x80000001 | 0x22d3 |
| 0.0.0.20      | 2.2.2.2       | 393 | 0x8000000d | 0xdd0f |

Inter-Area-Prefix-LSA (Area 0.0.0.1)

| Link State ID | Origin Router | Age | Seq#       | CkSum  |
|---------------|---------------|-----|------------|--------|
| 0.0.0.1       | 2.2.2.2       | 552 | 0x8000000d | 0x6296 |

Intra-Area-Prefix-LSA (Area 0.0.0.1)

| Link State ID | Origin Router | Age | Seq#       | CkSum  | Prefix Reference |
|---------------|---------------|-----|------------|--------|------------------|
| 0.0.0.1       | 2.2.2.2       | 399 | 0x8000000d | 0x4f5c | 1 Network-LSA    |
| 0.0.0.2       | 2.2.2.2       | 244 | 0x80000001 | 0x6b4b | 1 Network-LSA    |

[~ABR] **display ospfv3 lsdb inter-prefix**

OSPFv3 Router with ID (2.2.2.2) (Process 1)

Inter-Area-Prefix-LSA (Area 0.0.0.0)

LS Age: 40  
LS Type: Inter-Area-Prefix-LSA  
Link State ID: 0.0.0.3  
Originating Router: 2.2.2.2

```
LS Seq Number: 0x80000001
Retransmit Count: 0
Checksum: 0x6dba
Length: 32

Metric: 1
Prefix: 2001:DB8::/32
Prefix Options: 0 (-|-|-|-)

Inter-Area-Prefix-LSA (Area 0.0.0.1)

LS Age: 562
LS Type: Inter-Area-Prefix-LSA
Link State ID: 0.0.0.1
Originating Router: 2.2.2.2
LS Seq Number: 0x8000000d
Retransmit Count: 0
Checksum: 0x6296
Length: 36

Metric: 1
Prefix: 2001:DB8:1::/64
Prefix Options: 0 (-|-|-|-)
```

# Run the **display ospfv3 abr-summary-list** command on the ABR. The command output shows information about route summarization in area 1.

```
[~ABR] display ospfv3 abr-summary-list

OSPFv3 Process (1)
Area ID : 0.0.0.1
Prefix Prefix-Len Matched Status
2001:DB8:: 32 1 [Active] Advertised
```

----End

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
ospfv3 1
router-id 3.3.3.3
area 0.0.0.1
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:DB8:2::1/64
ospfv3 1 area 0.0.0.1
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
ospfv3 1
router-id 1.1.1.1
area 0.0.0.0
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:DB8:1::2/64
ospfv3 1 area 0.0.0.0
```

- ```
#  
return
```
- Device C configuration file

```
#  
sysname DeviceC  
#  
ospfv3 1  
router-id 4.4.4.4  
area 0.0.0.1  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:DB8:3::1/64  
ospfv3 1 area 0.0.0.1  
#  
return
```
 - ABR configuration file

```
#  
sysname ABR  
#  
ospfv3 1  
router-id 2.2.2.2  
area 0.0.0.0  
area 0.0.0.1  
abr-summary 2001:DB8:: 32  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:DB8:1::1/64  
ospfv3 1 area 0.0.0.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:DB8:2::2/64  
ospfv3 1 area 0.0.0.1  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:DB8:3::2/64  
ospfv3 1 area 0.0.0.1  
#  
return
```

Example for Configuring an OSPFv3 NSSA

This section provides an example for configuring an OSPFv3 not-so-stubby area (NSSA).

Networking Requirements

An excessive number of entries in a routing table waste network resources and cause high CPU usage. To address the problem, configure a non-backbone area on the border of an AS as an NSSA. NSSAs can import AS external routes and advertise them within the entire OSPFv3 AS, without learning external routes from other areas in the AS, which reduces bandwidth and storage resource consumption on the router.

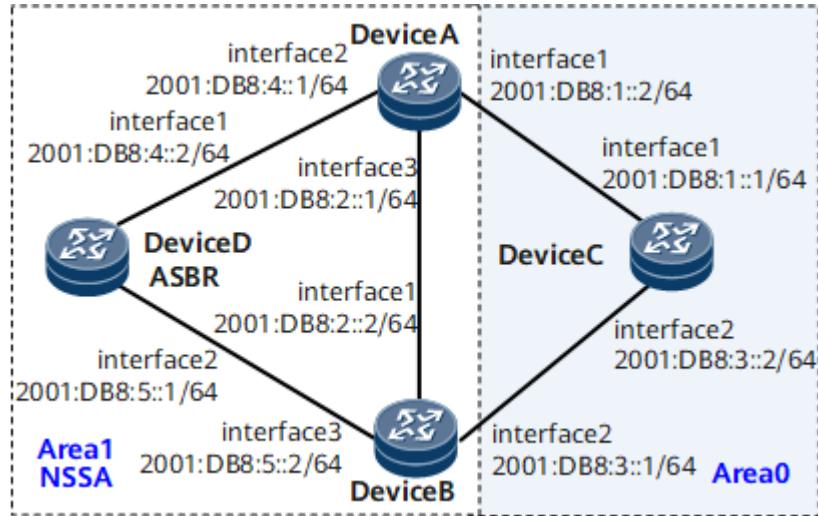
In [Figure 1-195](#), OSPFv3 runs on all routers, and the entire AS is partitioned into two areas. Device A and Device B function as ABRs to forward inter-area routes;

Device D functions as an ASBR and imports the external static route 2001:DB8:6::1/128. To import AS external routes but reduce the number of LSAs to be advertised to area 1 without affecting route reachability, configure area 1 as an NSSA.

Figure 1-195 Networking for configuring an OSPFv3 NSSA

 NOTE

Interfaces 1 through 3 in this example represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



Configuration Notes

To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPFv3 functions on all routers to ensure that they can communicate with each other using OSPFv3.
2. Configure area 1 as an NSSA.
3. Configure Device C to import the static route 2001:DB8:7::1/128.
4. Configure Device D to import the static route 2001:DB8:6::1/128.

Data Preparation

To complete the configuration, you need the following data:

- Router ID 1.1.1.1 of Device A, OSPFv3 process ID 1, network segment 2001:DB8:1::0/64 of area 0, and network segments 2001:DB8:2::0/64 and 2001:DB8:4::0/64 of area 1

- Router ID 2.2.2.2 of Device B, OSPFv3 process ID 1, network segment 2001:DB8:3::0/64 of area 0, and network segments 2001:DB8:2::0/64 and 2001:DB8:5::0/64 of area 1
- Router ID 3.3.3.3 of Device C, OSPFv3 process ID 1, and network segments 2001:DB8:1::0/64 and 2001:DB8:3::0/64 of area 0
- Router ID 4.4.4.4 of Device D, OSPFv3 process ID 1, and network segments 2001:DB8:4::0/64 and 2001:DB8:5::0/64 of area 1

Procedure

Step 1 Configure an IP address for each interface.

Configure an IP address to each interface according to [Figure 1-195](#). For details about the configuration, see Configuration Files in this section.

Step 2 Configure basic OSPFv3 functions.

For detailed operations, see [Example for Configuring Basic OSPFv3 Functions](#).

Step 3 Configure area 1 as an NSSA.

Configure Device A.

```
[~DeviceA] ospfv3
[*DeviceA-ospfv3-1] area 1
[*DeviceA-ospfv3-1-area-0.0.0.1] nssa
[*DeviceA-ospfv3-1-area-0.0.0.1] commit
[~DeviceA-ospfv3-1-area-0.0.0.1] quit
[~DeviceA-ospfv3-1] quit
```

Configure Device B.

```
[~DeviceB] ospfv3
[*DeviceB-ospfv3-1] area 1
[*DeviceB-ospfv3-1-area-0.0.0.1] nssa
[*DeviceB-ospfv3-1-area-0.0.0.1] commit
[~DeviceB-ospfv3-1-area-0.0.0.1] quit
[~DeviceB-ospfv3-1] quit
```

Configure Device D.

```
[~DeviceD] ospfv3
[*DeviceD-ospfv3-1] area 1
[*DeviceD-ospfv3-1-area-0.0.0.1] nssa
[*DeviceD-ospfv3-1-area-0.0.0.1] commit
[~DeviceD-ospfv3-1-area-0.0.0.1] quit
[~DeviceD-ospfv3-1] quit
```

Step 4 Configure Device C to import the static route 2001:DB8:7::1/128.

```
[~DeviceC] ipv6 route-static 2001:DB8:7::1 128 NULL0
[*DeviceC] ospfv3
[*DeviceC-ospfv3-1] import-route static
[*DeviceC-ospfv3-1] commit
[~DeviceC-ospfv3-1] quit
```

Step 5 Configure Device D to import the static route 2001:DB8:6::1/128.

```
[~DeviceD] ipv6 route-static 2001:DB8:6::1 128 NULL0
[*DeviceD] ospfv3
[*DeviceD-ospfv3-1] import-route static
[*DeviceD-ospfv3-1] commit
[~DeviceD-ospfv3-1] quit
```

Step 6 Verify the configuration.

Display the OSPFv3 routing tables on Device C and Device D.

```
[~DeviceC] display ospfv3 routing
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
        N - NSSA
Flags : A - Added to URT6, LT - Locator Routing

OSPFv3 Process (1)
Destination                               Metric
Next-hop
2001:DB8:1::/64                           1
    directly connected, Vlanif18, Flags : A
IA 2001:DB8:2::/64                         2
    via FE80::3A6D:7CFF:FE21:1200, Vlanif18, Flags : A
IA 2001:DB8:2::/64                         2
    via FE80::3A6D:7CFF:FE41:1200, Vlanif14, Flags : A
2001:DB8:3::/64                           1
    directly connected, Vlanif14, Flags : A
IA 2001:DB8:4::/64                         2
    via FE80::3A6D:7CFF:FE21:1200, Vlanif18, Flags : A
IA 2001:DB8:5::/64                         2
    via FE80::3A6D:7CFF:FE41:1200, Vlanif14, Flags : A
E2 2001:DB8:6::1/128                       1
    via FE80::3A6D:7CFF:FE41:1200, Vlanif14, Flags : A
[~DeviceD] display ospfv3 routing
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
        N - NSSA
Flags : A - Added to URT6, LT - Locator Routing

OSPFv3 Process (1)
Destination                               Metric
Next-hop
E2 ::/0                                    1
N  via FE80::3A6D:7CFF:FE21:1200, Vlanif13, Flags : A
E2 ::/0                                    1
N  via FE80::3A6D:7CFF:FE41:1200, Vlanif11, Flags : A
IA 2001:DB8:1::/64                         2
    via FE80::3A6D:7CFF:FE21:1200, Vlanif13, Flags : A
2001:DB8:2::/64                           2
    via FE80::3A6D:7CFF:FE21:1200, Vlanif13, Flags : A
2001:DB8:2::/64                           2
    via FE80::3A6D:7CFF:FE41:1200, Vlanif11, Flags : A
IA 2001:DB8:3::/64                         2
    via FE80::3A6D:7CFF:FE41:1200, Vlanif11, Flags : A
2001:DB8:4::/64                           1
    directly connected, Vlanif13, Flags : A
2001:DB8:5::/64                           1
    directly connected, Vlanif11, Flags : A
```

The command output shows that Device C has imported an AS external route (2001:DB8:6::1/128) and that the router that advertises this route is DeviceB. In addition, the NSSA does not learn the 2001:DB8:7::1/128 route from area 0.

Display routing information of the NSSA on routers. In the following example, the command output on Device A is used.

```
[~DeviceA] display ospfv3 routing nssa-routes
Codes : E2 - Type 2 External, E1 - Type 1 External, IA - Inter-Area,
        N - NSSA
Flags : A - Added to URT6

OSPFv3 Process (1)
Destination                               Metric
Next-hop
E2 2001:DB8:6::1/128                     1
N  via FE80::3A6D:7CFF:FE11:1200, Vlanif13, Flags : A
```

Display NSSA LSA information on routers. In the following example, the command output on Device A is used.

```
[~DeviceA] display ospfv3 lsdb nssa
    OSPFv3 Router with ID (1.1.1.1) (Process 1)

        NSSA-external-LSA (Area 0.0.0.1)

            LS Age: 391
            LS Type: NSSA-external-LSA
            Link State ID: 0.0.0.0
            Originating Router: 1.1.1.1
            LS Seq Number: 0x80000001
            Retransmit Count: 0
            Checksum: 0x6ebe
            Length: 32
            Flags: (E|-|T)
            Metric Type: 2 (Larger than any link state path)
                Metric: 1
            Prefix: ::/0
            Prefix Options: 0 (-|-|-|-)
            Tag: 0

            LS Age: 378
            LS Type: NSSA-external-LSA
            Link State ID: 0.0.0.0
            Originating Router: 2.2.2.2
            LS Seq Number: 0x80000001
            Retransmit Count: 0
            Checksum: 0x50d8
            Length: 32
            Flags: (E|-|T)
            Metric Type: 2 (Larger than any link state path)
                Metric: 1
            Prefix: ::/0
            Prefix Options: 0 (-|-|-|-)
            Tag: 0

            LS Age: 429
            LS Type: NSSA-external-LSA
            Link State ID: 0.0.0.1
            Originating Router: 4.4.4.4
            LS Seq Number: 0x80000001
            Retransmit Count: 0
            Checksum: 0xb7e0
            Length: 48
            Flags: (E|-|T)
            Metric Type: 2 (Larger than any link state path)
                Metric: 1
            Prefix: 2001:DB8:6::1/128
            Prefix Options: 8 (-|P|-|-)
```

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA  
#  
vlan batch 13 16 18  
#  
ospfv3 1  
router-id 1.1.1.1  
area 0.0.0.0  
area 0.0.0.1  
nssa  
#  
interface Vlanif13  
ipv6 enable  
ipv6 address 2001:DB8:4::1/64
```

```
ospfv3 1 area 0.0.0.1
#
interface Vlanif16
    ipv6 enable
    ipv6 address 2001:DB8:2::1/64
    ospfv3 1 area 0.0.0.1
#
interface Vlanif18
    ipv6 enable
    ipv6 address 2001:DB8:1::2/64
    ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet1/0/0
    undo shutdown
    port default vlan 18
#
interface GigabitEthernet2/0/0
    undo shutdown
    port default vlan 13
#
interface GigabitEthernet3/0/0
    undo shutdown
    port default vlan 16
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
vlan batch 11 14 16
#
ospfv3 1
    router-id 2.2.2.2
    area 0.0.0.0
    area 0.0.0.1
    nssa
#
interface Vlanif11
    ipv6 enable
    ipv6 address 2001:DB8:5::2/64
    ospfv3 1 area 0.0.0.1
#
interface Vlanif14
    ipv6 enable
    ipv6 address 2001:DB8:3::1/64
    ospfv3 1 area 0.0.0.0
#
interface Vlanif16
    ipv6 enable
    ipv6 address 2001:DB8:2::2/64
    ospfv3 1 area 0.0.0.1
#
interface GigabitEthernet1/0/0
    undo shutdown
    port default vlan 16
#
interface GigabitEthernet2/0/0
    undo shutdown
    port default vlan 14
#
interface GigabitEthernet3/0/0
    undo shutdown
    port default vlan 11
#
return
```

- Device C configuration file

```
#
sysname DeviceC
```

```
#  
vlan batch 14 18  
#  
ospfv3 1  
router-id 3.3.3.3  
import-route static  
area 0.0.0.0  
#  
interface Vlanif14  
ipv6 enable  
ipv6 address 2001:DB8:3::2/64  
ospfv3 1 area 0.0.0.0  
#  
interface Vlanif18  
ipv6 enable  
ipv6 address 2001:DB8:1::1/64  
ospfv3 1 area 0.0.0.0  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
port default vlan 18  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
port default vlan 14  
#  
ipv6 route-static 2001:DB8:7::1 128 NULL0  
#  
return
```

- Device D configuration file

```
#  
sysname DeviceD  
#  
vlan batch 11 13  
#  
ospfv3 1  
router-id 4.4.4.4  
import-route static  
area 0.0.0.1  
nssa  
#  
interface Vlanif11  
ipv6 enable  
ipv6 address 2001:DB8:5::1/64  
ospfv3 1 area 0.0.0.1  
#  
interface Vlanif13  
ipv6 enable  
ipv6 address 2001:DB8:4::2/64  
ospfv3 1 area 0.0.0.1  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
port default vlan 13  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
port default vlan 11  
#  
ipv6 route-static 2001:DB8:6::1 128 NULL0  
#  
return
```

Example for Configuring OSPFv3 Delay Reporting to BGP-LS

OSPFv3 delay is reported to BGP-LS, which summarizes the topology information collected by IGPs and sends the information to the upper-layer controller for delay-based path computation.

Networking Requirements

BGP-LS is a new method of collecting network topology information. The topology information discovered by IGPs is summarized and reported to an upper-layer controller through BGP. With powerful routing capabilities of BGP, BGP-LS has the following advantages:

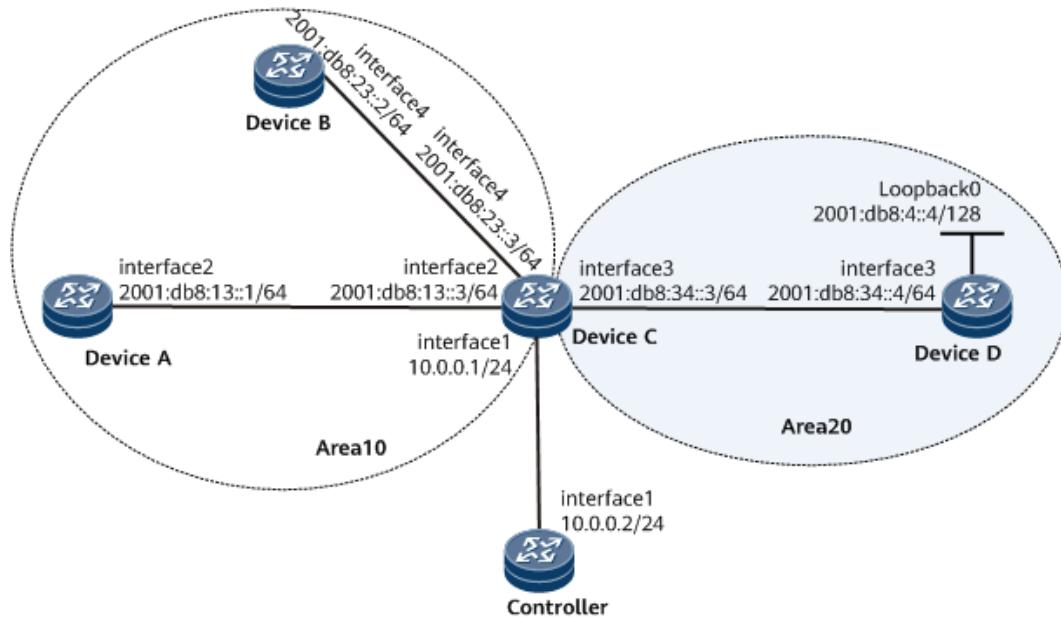
- Lowers the requirements on the controller's computing capabilities and no longer requires IGP capabilities on the controller.
- Facilitates route selection and computation on the controller by using BGP to summarize process or AS topology information and report the complete information to the controller.
- Requires only one routing protocol (BGP) to report topology information to the controller.

In **Figure 1-196**, DeviceC is connected to the controller and reports topology information to the controller. DeviceA, DeviceB, DeviceC, and DeviceD use OSPFv3 to implement IP network interworking. The two interfaces connecting DeviceC and DeviceD belong to area 20, and the interfaces connecting DeviceA, DeviceB, and DeviceC belong to area 10.

Figure 1-196 Configuring BGP-LS

 NOTE

interface1, interface2, interface3, and interface4 in this example represent GigabitEthernet1/0/1, GigabitEthernet1/0/2, GigabitEthernet1/0/3, and GigabitEthernet1/0/4, respectively.



Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IPv6 addresses for interfaces on each device.
2. Configure basic OSPFv3 functions.
3. Deploy BGP-LS on DeviceC and the controller.
4. Configure the TWAMP Light Controller and TWAMP Light Responder on DeviceA and DeviceC, respectively.
5. Configure the TE function.
6. Configure delay advertisement on DeviceA.

Data Preparation

To complete the configuration, you need the following data:

- IPv6 addresses and areas of interfaces on DeviceA, DeviceB, DeviceC, and DeviceD
- BGP-LS ID in OSPFv3 on DeviceC
- BGP AS numbers, BGP-LS domain AS numbers, and BGP-LS domain IDs of Device C and the controller

Procedure

Step 1 Assign an IPv6 address to each interface on each device. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic OSPFv3 functions.

Configure DeviceA.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] area 0.0.0.10
[*DeviceA-ospfv3-1-area-0.0.0.10] quit
[*DeviceA-ospfv3-1] quit
[*DeviceA] interface gigabitethernet 1/0/2
[*DeviceA-GigabitEthernet1/0/2] ospfv3 1 area 0.0.0.10
[*DeviceA-GigabitEthernet1/0/2] ospfv3 network-type p2p
[*DeviceA-GigabitEthernet1/0/2] commit
[~DeviceA-GigabitEthernet1/0/2] quit
```

Configure DeviceB.

```
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] router-id 2.2.2.2
[*DeviceB-ospfv3-1] area 0.0.0.10
[*DeviceB-ospfv3-1-area-0.0.0.10] quit
[*DeviceB-ospfv3-1] quit
[*DeviceB] interface gigabitethernet 1/0/4
[*DeviceB-GigabitEthernet1/0/4] ospfv3 1 area 0.0.0.10
[*DeviceB-GigabitEthernet1/0/4] commit
```

```
[~DeviceB-GigabitEthernet1/0/4] quit
# Configure DeviceC.

[~DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] router-id 3.3.3.3
[*DeviceC-ospfv3-1] area 0.0.0.10
[*DeviceC-ospfv3-1-area-0.0.0.10] quit
[*DeviceC-ospfv3-1] area 0.0.0.20
[*DeviceC-ospfv3-1-area-0.0.0.20] quit
[*DeviceC-ospfv3-1] quit
[*DeviceC] interface gigabitethernet 1/0/2
[*DeviceC-GigabitEthernet1/0/2] ospfv3 1 area 0.0.0.10
[*DeviceC-GigabitEthernet1/0/2] ospfv3 network-type p2p
[*DeviceC-GigabitEthernet1/0/2] quit
[*DeviceC] interface gigabitethernet 1/0/3
[*DeviceC-GigabitEthernet1/0/3] ospfv3 1 area 0.0.0.20
[*DeviceC-GigabitEthernet1/0/3] quit
[*DeviceC] interface gigabitethernet 1/0/4
[*DeviceC-GigabitEthernet1/0/4] ospfv3 1 area 0.0.0.10
[*DeviceC-GigabitEthernet1/0/4] commit
[~DeviceC-GigabitEthernet1/0/4] quit
```

Configure DeviceD.

```
[~DeviceD] ospfv3 1
[*DeviceD-ospfv3-1] router-id 4.4.4.4
[*DeviceD-ospfv3-1] area 0.0.0.20
[*DeviceD-ospfv3-1-area-0.0.0.20] quit
[*DeviceD-ospfv3-1] quit
[*DeviceD] interface gigabitethernet 1/0/3
[*DeviceD-GigabitEthernet1/0/3] ospfv3 1 area 0.0.0.20
[*DeviceD-GigabitEthernet1/0/3] quit
[*DeviceD] interface LoopBack0
[*DeviceD-LoopBack0] ospfv3 1 area 0.0.0.20
[*DeviceD-LoopBack0] commit
[~DeviceD-LoopBack0] quit
```

Step 3 Deploy BGP-LS on DeviceC and the controller.

Enable OSPFv3 topology advertisement on DeviceC.

```
[~DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] bgp-ls enable
[*DeviceC-ospfv3-1] bgp-ls identifier 20
[*DeviceC-ospfv3-1] commit
[~DeviceC-ospfv3-1] quit
```

Enable BGP-LS on DeviceC and establish a BGP-LS peer relationship with the controller.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] peer 10.0.0.2 as-number 100
[*DeviceC-bgp] link-state-family unicast
[*DeviceC-bgp-af-ls] peer 10.0.0.2 enable
[*DeviceC-bgp-af-ls] commit
[~DeviceC-bgp-af-ls] quit
[~DeviceC-bgp] quit
```

Enable BGP-LS on the controller and establish a BGP-LS peer relationship with DeviceC.

```
[~Controller] bgp 100
[*Controller-bgp] peer 10.0.0.1 as-number 100
[*Controller-bgp] link-state-family unicast
[*Controller-bgp-af-ls] peer 10.0.0.1 enable
[*Controller-bgp-af-ls] commit
[~Controller-bgp-af-ls] quit
[~Controller-bgp] quit
```

Step 4 Configure the TWAMP Light Controller and TWAMP Light Responder on DeviceA and DeviceC, respectively.

Configure the TWAMP Light Controller on DeviceA.

```
[~DeviceA] nqa twamp-light
[*DeviceA-twamp-light] client
[*DeviceA-twamp-light-client] test-session 1 sender-ipv6 2001:db8:13::1 reflector-ipv6 2001:db8:13::3
sender-port 862 reflector-port 862
[*DeviceA-twamp-light-client] test-session 1 bind interface GigabitEthernet 1/0/2
[*DeviceA-twamp-light-client] quit
[*DeviceA-twamp-light] sender
[*DeviceA-twamp-light-sender] test start-continual test-session 1 period 1000
[*DeviceA-twamp-light-sender] commit
[~DeviceA-twamp-light-sender] quit
[~DeviceA-twamp-light] quit
```

Configure the TWAMP Light Responder on DeviceC.

```
[~DeviceC] nqa twamp-light
[*DeviceC-twamp-light] responder
[*DeviceC-twamp-light-responder] test-session 1 local-ipv6 2001:db8:13::3 remote-ipv6 2001:db8:13::1
local-port 862 remote-port 862
[*DeviceC-twamp-light-responder] commit
[~DeviceC-twamp-light-responder] quit
[~DeviceC-twamp-light] quit
```

Step 5 Configure TE on DeviceA.

```
[~DeviceA] te attribute enable
[*DeviceA] commit
```

Step 6 Enable TE for the OSPFv3 area on DeviceA.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] area 0.0.0.10
[*DeviceA-ospfv3-1-area-0.0.0.10] traffic-eng enable
[*DeviceA-ospfv3-1-area-0.0.0.10] commit
```

Step 7 Configure delay advertisement on DeviceA.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospf-1] metric-delay advertisement enable
[*DeviceA-ospf-1] commit
```

Step 8 Verify the configuration.

Display information about BGP-LS peers and their status on DeviceC.

```
[~DeviceC] display bgp link-state unicast peer
BGP local router ID : 10.0.0.1
Local AS number : 100
Total number of peers : 1          Peers in established state : 1
Peer          V      AS  MsgRcvd  MsgSent  OutQ  Up/Down  State  PrefRcv
10.0.0.2      4      100   5       9       0 00:00:55 Established   0
```

Check the delay information advertised by OSPFv3 on DeviceC.

```
[~DeviceC] display ospfv3 traffic-eng
OSPFV3 Process 1 with Router ID 3.3.3.3

Area ID          : 0.0.0.10
Traffic Engineering LSAs of the database
-----
LSA [1]
Last Receive TimeStamp : 0000-00-00 00:00:00
Last Update TimeStamp : 2023-04-21 01:27:50
```

```
-----  
Lsa Type      : Intra-Area-TE  
Link State Id : 0.0.0  
Advertising Router Id : 1.1.1.1  
Length        : 40  
  
Router IPv6 Address : 2001:DB8:1::1  
  
-----  
LSA [2]  
Last Receive TimeStamp : 0000-00-00 00:00:00  
Last Update TimeStamp : 2023-04-21 01:28:22  
-----  
Lsa Type      : Intra-Area-TE  
Link State Id : 0.0.0.1  
Advertising Router Id : 1.1.1.1  
Length        : 160  
Link Type     : P2P  
Neighbor Interface Id : 14  
Neighbor Router Id : 3.3.3.3  
Local Interface IPv6 Address : 2001:db8:13::1  
TE Metric     : 1  
Maximum Bandwidth : 12500000 bytes/sec  
Maximum Reservable BW : 0 bytes/sec  
Admin Group   : 0x0  
Global Pool   :  
    Unreserved BW [ 0] = 0 bytes/sec  
    Unreserved BW [ 1] = 0 bytes/sec  
    Unreserved BW [ 2] = 0 bytes/sec  
    Unreserved BW [ 3] = 0 bytes/sec  
    Unreserved BW [ 4] = 0 bytes/sec  
    Unreserved BW [ 5] = 0 bytes/sec  
    Unreserved BW [ 6] = 0 bytes/sec  
    Unreserved BW [ 7] = 0 bytes/sec  
Min/Max Unidirectional Link Delay:  
Anomalous (A) Bit: 0  
Min Delay: 2220 (us)  
Max Delay: 4240 (us)
```

Check BGP-LS routes on DeviceC.

```
[~DeviceC] display bgp link-state unicast routing-table [LINK][OSPFv3][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.0.0.1][ospf-area-id0.0.0.10][igp-router-id1.1.1.1]][REMOTE[as100][bgp-ls-identifier10.0.0.1][ospf-area-id0.0.0.10][igp-router-id3.3.3.3]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address2001:db8:13::1][peer-address2001:db8:13::3]]]  
BGP Local router ID is 10.1.1.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
              h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
  
BGP local router ID : 3.3.3.3  
Local AS number : 100  
Paths: 1 available, 1 best, 1 select  
BGP routing table entry information of [LINK][OSPFv3][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.0.0.1][ospf-area-id0.0.0.10][igp-router-id1.1.1.1]][REMOTE[as100][bgp-ls-identifier10.0.0.1][ospf-area-id0.0.0.10][igp-router-id3.3.3.3]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address2001:db8:13::1][peer-address2001:db8:13::3]]:  
Imported route.  
From: 0.0.0.0 (0.0.0.0)  
Route Duration: 0d00h09m59s  
Direct Out-interface:  
Original nexthop: 0.0.0.0  
AS-path Nil, origin incomplete, MED 0, pref-val 0, valid, local, best, select, pre 255  
Link-state Route Type: LINK  
Protocol: OSPFv3  
Identifier: 20  
Local Node Descriptor:  
AS Number: 100  
BGP Identifier: 10.0.0.1
```

```
OSPF Area ID: 0.0.0.10
IGP Router ID: 1.1.1.1
Remote Node Descriptor:
AS Number: 100
BGP Identifier: 10.0.0.1
OSPF Area ID: 0.0.0.10
IGP Router ID: 3.3.3.3
Link Descriptor:
IPv4 interface address: 0.0.0.0
IPv4 neighbor address: 0.0.0.0
IPv6 interface address: 2001:db8:13::1
IPv6 neighbor address: 2001:db8:13::3
Link-state attribute:
IPv4 Router ID of Local Node:
IPv6 Router ID of Local Node:
IPv4 Router ID of Remote Node:
IPv6 Router ID of Remote Node:
Administrative group: 0x0
Maximum link bandwidth(kbits/sec): 0
Maximum reservable link bandwidth(kbits/sec): 0
Maximum Unreserved bandwidth(kbits/sec): 0 0 0 0 0 0 0 0
TE Default Metric: 0
Link Protection Type: Null
MPLS Protocol Mask: 0x0
IGP Metric: 1
Shared Risk Link Group:
Min/Max Unidirectional Link Delay(microseconds): 912/5503
Opaque link Properties:
Link Name:
Adjacency Segment Identifier(Flags/Weight/SID):
LAN Adjacency Segment Identifier(Flags/Weight/System-ID/SID):
SRv6 End.X SID(EndPoint Behavior/Flags/Algorithm/Weight/SID)(SID Structure):
SRv6 LAN End.X SID(EndPoint Behavior/Flags/Algorithm/Weight/Neighbor ID/SID)(SID Structure):
MSD([MSD Type, MSD Value]):
Advertised to such 1 peers:
10.0.0.2
```

The preceding command output shows that DeviceC obtains the topology information on the whole OSPFv3 network. DeviceC can use BGP-LS routes to report the topology information to its BGP-LS peer (the controller).

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA  
#  
te attribute enable  
#  
ospfv3 1  
router-id 1.1.1.1  
metric-delay advertisement enable  
area 0.0.0.10  
traffic-eng enable  
#  
interface GigabitEthernet1/0/2  
undo shutdown  
ipv6 enable  
ipv6 address 2001:db8:13::1/64  
ospfv3 network-type p2p  
ospfv3 1 area 0.0.0.10  
#  
nqa twamp-light  
client  
test-session 1 sender-ipv6 2001:db8:13::1 reflector-ipv6 2001:db8:13::3 sender-port 862 reflector-port  
862
```

```
test-session 1 bind interface GigabitEthernet 1/0/2
sender
test start-continual test-session 1 period 1000
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
ospfv3 1
router-id 2.2.2.2
area 0.0.0.10
#
interface GigabitEthernet1/0/4
undo shutdown
ipv6 enable
ipv6 address 2001:db8:23::2/64
ospfv3 1 area 0.0.0.10
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
ospfv3 1
router-id 3.3.3.3
bgp-ls enable
bgp-ls identifier 20
area 0.0.0.10
area 0.0.0.20
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.0.0.1 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:13::3/64
ospfv3 network-type p2p
ospfv3 1 area 0.0.0.10
#
interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:34::3/64
ospfv3 1 area 0.0.0.20
#
interface GigabitEthernet1/0/4
undo shutdown
ipv6 enable
ipv6 address 2001:db8:23::3/64
ospf 1 area 0.0.0.10
#
bgp 100
peer 10.0.0.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 10.0.0.2 enable
#
link-state-family unicast
peer 10.0.0.2 enable
#
nqa twamp-light
responder
test-session 1 local-ipv6 2001:db8:13::3 remote-ipv6 2001:db8:13::1 local-port 862 remote-port 862
```

- ```

return
```
- DeviceD configuration file

```

sysname DeviceD

ospfv3 1
router-id 4.4.4.4
area 0.0.0.20

interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:34::4/64
ospfv3 1 area 0.0.0.20

interface LoopBack0
ipv6 enable
ipv6 address 2001:db8:4::4/128
ospfv3 1 area 0.0.0.20

return
```
  - Controller configuration file

```

sysname Controller

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.0.0.2 255.255.255.0

bgp 100
peer 10.0.0.1 as-number 100

ipv4-family unicast
undo synchronization
peer 10.0.0.1 enable

link-state-family unicast
peer 10.0.0.1 enable

return
```

## Example for Configuring Routing Loop Detection for OSPFv3 Multi-Process Mutual Route Import

This section provides an example for configuring routing loop detection for OSPFv3 multi-process mutual route import.

### Networking Requirements

On the live network, routes of an OSPFv3 process can be imported to another OSPFv3 process for redistribution. In such a scenario, routing policies are usually configured on multiple devices to prevent routing loops. If routing policies are incorrectly configured on the devices that import routes, routing loops may occur. To prevent this problem, configure routing loop detection for the routes imported to OSPFv3.

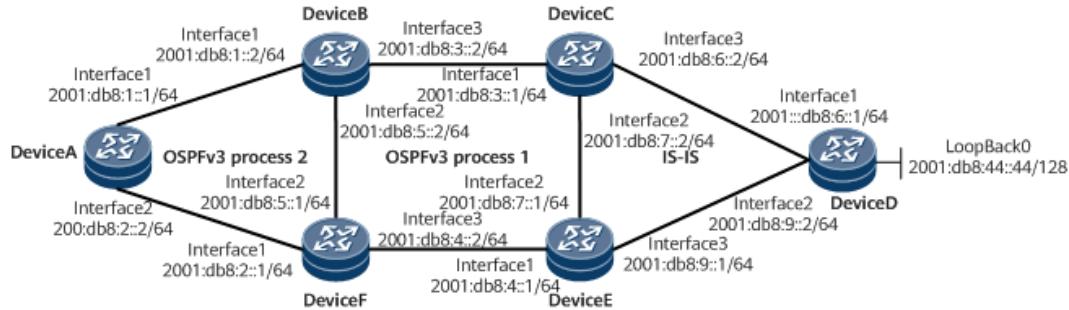
In [Figure 1-197](#), an IS-IS process is configured on DeviceC, DeviceD, and DeviceE. OSPFv3 process 1 is configured on DeviceB, DeviceC, DeviceE, and DeviceF. OSPFv3 process 2 is configured on DeviceA, DeviceB, and DeviceF. DeviceC is configured to import routes from the IS-IS process to OSPFv3 process 1. DeviceB is configured to

import routes from OSPFv3 process 1 to OSPFv3 process 2. DeviceF is configured to import routes from OSPFv3 process 2 to OSPFv3 process 1.

**Figure 1-197** Routing loop detection for OSPFv3 multi-process mutual route import

 **NOTE**

Interfaces 1, 2, and 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



## Precautions

To improve security, you are advised to deploy OSPFv3 authentication. For details, see "Configuring OSPFv3 Authentication". OSPFv3 IPSec is used as an example. For details, see "Example for Configuring IPSec for OSPFv3".

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IP addresses for interfaces on each device.
2. Enable IS-IS and OSPFv3 and configure basic functions.
3. Configure route import to construct a routing loop.
4. Check whether a routing loop occurs.
5. Enable routing loop detection to check whether the routing loop is eliminated.

## Procedure

**Step 1** Assign an IP address to each interface. Take DeviceA as an example.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ipv6 enable
[*DeviceA-GigabitEthernet1/0/0] ipv6 address 2001:db8:1::1 64
[*DeviceA-GigabitEthernet1/0/0] ipv6 address FE80::A:B link-local
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] ipv6 enable
[*DeviceA-GigabitEthernet2/0/0] ipv6 address 2001:db8:2::2 64
[*DeviceA-GigabitEthernet2/0/0] ipv6 address FE80::A:F link-local
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

The configurations of other devices are similar to those of DeviceA. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Enable OSPFv3 and IS-IS, and configure basic OSPFv3 and IS-IS functions to implement intra-area interworking.

# Configure an IS-IS process on DeviceC, DeviceD, and DeviceE. DeviceC is used as an example.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] cost-style wide
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] ipv6 enable topology standard
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] interface gigabitethernet 3/0/0
[*DeviceC-GigabitEthernet3/0/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet3/0/0] quit
[*DeviceC] commit
```

# Configure OSPFv3 process 1 on DeviceB, DeviceC, DeviceE, and DeviceF. DeviceB is used as an example.

```
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] router-id 2.2.2.1
[*DeviceB-ospfv3-1] area 0
[*DeviceB-ospfv3-1-area-0.0.0] quit
[*DeviceB-ospfv3-1] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface gigabitethernet 3/0/0
[*DeviceB-GigabitEthernet3/0/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet3/0/0] quit
[*DeviceB] commit
```

# Configure OSPFv3 process 2 on DeviceA, DeviceB, and DeviceF. DeviceA is used as an example.

```
[~DeviceA] ospfv3 2
[*DeviceA-ospfv3-2] router-id 1.1.1.2
[*DeviceA-ospfv3-2] area 0
[*DeviceA-ospfv3-2-area-0.0.0] quit
[*DeviceA-ospfv3-2] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ospfv3 2 area 0.0.0.0
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] ospfv3 2 area 0.0.0.0
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

**Step 3** Configure route import.

# Configure OSPFv3 process 1 on DeviceC and DeviceE to import routes from IS-IS. DeviceC is used as an example.

```
[~DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] router-id 3.3.3.1
[*DeviceC-ospfv3-1] import-route isis 1
[*DeviceC-ospfv3-1] quit
[*DeviceC] commit
```

# Configure OSPFv3 process 2 on DeviceB to import routes from OSPFv3 process 1.

```
[~DeviceB] ospfv3 2
[*DeviceB-ospfv3-2] router-id 2.2.2.2
[*DeviceB-ospfv3-2] import-route ospfv3 1
```

```
[*DeviceB-ospfv3-2] quit
[*DeviceB] commit
```

# Configure OSPFv3 process 1 on DeviceF to import routes from OSPFv3 process 2.

```
[~DeviceF] ospfv3 1
[*DeviceF-ospfv3-1] router-id 6.6.6.1
[*DeviceF-ospfv3-1] import-route ospfv3 2
[*DeviceF-ospfv3-1] quit
[*DeviceF] commit
```

**Step 4** Display the routing table on each device to check whether a routing loop occurs.

# Display OSPFv3 neighbor information on DeviceB.

```
[~DeviceB] display ospfv3 peer
OSPFv3 Process (1)
Total number of peer(s): 2
Peer(s) in full state: 2
OSPFv3 Area (0.0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
6.6.6.1 1 Full/DR 00:00:31 GE2/0/0 0
3.3.3.1 1 Full/DR 00:00:36 GE3/0/0 0

OSPFv3 Process (2)
Total number of peer(s): 1
Peer(s) in full state: 1
OSPFv3 Area (0.0.0.0)
Neighbor ID Pri State Dead Time Interface Instance ID
1.1.1.2 1 Full/Backup 00:00:36 GE1/0/0 0
```

The command output shows that OSPFv3 neighbor relationships have been established between devices.

# Display the routing table on DeviceB.

```
[~DeviceB] display ipv6 routing-table 2001:db8:44::44
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 2

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::F:B Preference : 150
Cost : 1 Protocol : OSPFv3ASE
RelayNextHop : :: TunnelID : 0x0
Interface : GE2/0/0 Flags : D

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::C:B Preference : 150
Cost : 1 Protocol : OSPFv3ASE
RelayNextHop : :: TunnelID : 0x0
Interface : GE3/0/0 Flags : D
```

The preceding routing table on DeviceB shows that one route with the next hop of DeviceF and the other route with the next hop of DeviceC implement load balancing.

# Display the routing table on DeviceA.

```
[~DeviceA] display ipv6 routing-table 2001:db8:44::44
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::B:A Preference : 150
Cost : 1 Protocol : OSPFv3ASE
```

```
RelayNextHop : :: TunnelID : 0x0
Interface : GE1/0/0 Flags : D
```

The preceding command output shows that the next hop of the route displayed on DeviceA is DeviceB.

# Display the routing table on DeviceF.

```
[~DeviceF] display ipv6 routing-table 2001:db8:44:44
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 2

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::E:F Preference : 150
Cost : 1 Protocol : OSPFv3ASE
RelayNextHop : :: TunnelID : 0x0
Interface : GE3/0/0 Flags : D

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::A:F Preference : 150
Cost : 1 Protocol : OSPFv3ASE
RelayNextHop : :: TunnelID : 0x0
Interface : GE1/0/0 Flags : D
```

The preceding routing table on DeviceF shows that one route with the next hop of DeviceA and the other route with the next hop of DeviceE implement load balancing.

This means that a routing loop occurs among DeviceB, DeviceA, and DeviceF.

**Step 5** Enable routing loop detection on each device.

# Enable routing loop detection for routes imported to OSPFv3. DeviceA is used as an example.

```
[~DeviceA] route loop-detect ospfv3 enable
[*DeviceA] commit
```

**Step 6** Check whether the routing loop is eliminated.

# Display the routing table on DeviceB.

```
[~DeviceB] display ipv6 routing-table 2001:db8:44:44
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::C:B Preference : 150
Cost : 1 Protocol : OSPFv3ASE
RelayNextHop : :: TunnelID : 0x0
Interface : GE3/0/0 Flags : D
```

The preceding command output shows that the next hop of the route displayed on DeviceB is DeviceC.

# Display the routing table on DeviceA.

```
[~DeviceA] display ipv6 routing-table 2001:db8:44:44
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:44::44 PrefixLength : 64
```

|                     |                      |
|---------------------|----------------------|
| NextHop : FE80::B:A | Preference : 150     |
| Cost : 1            | Protocol : OSPFv3ASE |
| RelayNextHop : ::   | TunnelID : 0x0       |
| Interface : GE1/0/0 | Flags : D            |

The preceding command output shows that the next hop of the route displayed on DeviceA is DeviceB.

# Display the routing table on DeviceF.

```
[~DeviceF] display ipv6 routing-table 2001:db8:44::44
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination : 2001:db8:44::44 PrefixLength : 64
NextHop : FE80::E:F Preference : 150
Cost : 1 Protocol : OSPFv3ASE
RelayNextHop : :: TunnelID : 0x0
Interface : GE3/0/0 Flags : D
```

The preceding command output shows that the next hop of the route displayed on DeviceF is DeviceE. This means that the routing loop on DeviceB, DeviceA, and DeviceF is eliminated.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

ospfv3 2
router-id 1.1.1.2
area 0.0.0.

route loop-detect ospfv3 enable

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
ipv6 address FE80::A:B link-local
ospfv3 2 area 0.0.0.0

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
ipv6 address FE80::A:F link-local
ospfv3 2 area 0.0.0.0

return
```

- DeviceB configuration file

```

sysname DeviceB

ospfv3 1
router-id 2.2.2.1
area 0.0.0.

ospfv3 2
router-id 2.2.2.2
import-route ospfv3 1
```

```
area 0.0.0
#
route loop-detect ospfv3 enable
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
ipv6 address FE80::B:A link-local
ospfv3 2 area 0.0.0
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::2/64
ipv6 address FE80::B:F link-local
ospfv3 1 area 0.0.0
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::2/64
ipv6 address FE80::B:C link-local
ospfv3 1 area 0.0.0
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
isis 1
is-level level-1
cost-style wide
network-entity 10.0000.0000.0003.00
#
ipv6 enable topology standard
#
#
ospfv3 1
router-id 3.3.3.1
import-route isis 1
area 0.0.0
#
route loop-detect ospfv3 enable
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
ipv6 address FE80::C:B link-local
ospfv3 1 area 0.0.0
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:7::2/64
ospfv3 1 area 0.0.0
isis ipv6 enable 1
#
interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:6::2/64
isis ipv6 enable 1
#
return
```

- DeviceD configuration file

```

sysname DeviceD

isis 1
is-level level-1
cost-style wide
network-entity 10.0000.0000.0004.00

ipv6 enable topology standard

route loop-detect ospfv3 enable

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:6::1/64
isis ipv6 enable 1

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9::2/64
isis ipv6 enable 1

interface LoopBack0
ipv6 enable
ipv6 address 2001:db8:44::44/128
isis ipv6 enable 1

return
```

- DeviceE configuration file

```

sysname DeviceE

isis 1
is-level level-1
cost-style wide
network-entity 10.0000.0000.0005.00

ipv6 enable topology standard

ospfv3 1
router-id 5.5.5.1
import-route isis 1
area 0.0.0.

route loop-detect ospfv3 enable

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
ipv6 address FE80::E:F link-local
ospfv3 1 area 0.0.0.0

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:7::1/64
ospfv3 1 area 0.0.0.0
isis ipv6 enable 1

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9::1/64
isis ipv6 enable 1
```

```

return
● DeviceF configuration file

sysname DeviceF

ospfv3 1
router-id 6.6.6.1
import-route ospfv3 2
area 0.0.0.0

ospfv3 2
router-id 6.6.6.2
area 0.0.0.0

route loop-detect ospfv3 enable

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
ipv6 address FE80::F:A link-local
ospfv3 2 area 0.0.0.0

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::1/64
ipv6 address FE80::F:B link-local
ospfv3 1 area 0.0.0.0

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::2/64
ospfv3 1 area 0.0.0.0

return
```

## 1.1.6 RIP Configuration

### 1.1.6.1 RIP Description

#### 1.1.6.1.1 Overview of RIP

##### Definition

Routing Information Protocol (RIP) is a simple Interior Gateway Protocol (IGP). RIP is used in small-scale networks, such as campus networks and simple regional networks.

As a distance-vector routing protocol, RIP exchanges routing information through User Datagram Protocol (UDP) packets with port number 520.

RIP employs the hop count as the metric to measure the distance to the destination. In RIP, by default, the number of hops from the router to its directly connected network is 0; the number of hops from the router to a network that is reachable through another router is 1, and so on. The hop count (the metric) equals the number of routers along the path from the local network to the destination network. To speed up route convergence, RIP defines the hop count as

an integer that ranges from 0 to 15. A hop count that is greater than or equal to 16 is classified as infinite, indicating that the destination network or host is unreachable. Due to the hop limit, RIP is not applicable to large-scale networks.

RIP has two versions:

- RIP version 1 (RIP-1), a classful routing protocol
- RIP version 2 (RIP-2), a classless routing protocol

RIP supports split horizon, poison reverse, and triggered update, which improves the performance and prevents routing loops.

## Purpose

As the earliest IGP, RIP is used in small and medium-sized networks. Its implementation is simple, and the configuration and maintenance of RIP are easier than those of Open Shortest Path First (OSPF) and Intermediate System-to-Intermediate System (IS-IS). Therefore, RIP is widely used on live networks.

### 1.1.6.1.2 Understanding RIP

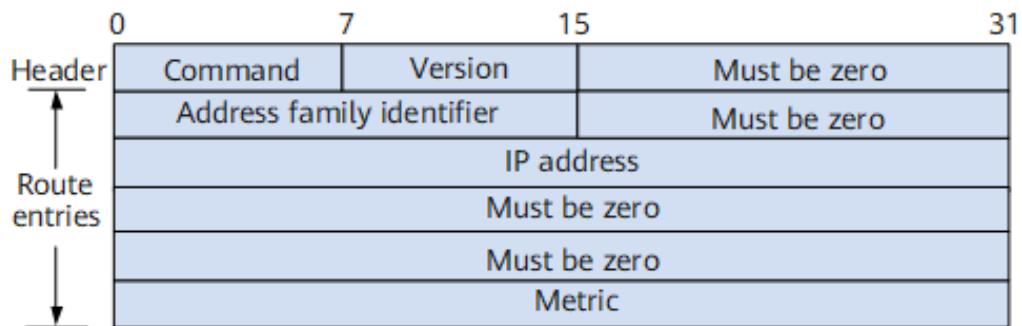
RIP is a distance-vector routing protocol. It forwards packets through UDP and uses timers to control the advertisement, update, and aging of routing information. However, design defects in RIP may cause routing loops. Therefore, split horizon, poison reverse, and triggered update were introduced into RIP to avoid routing loops.

In addition, RIP periodically advertises its routing table to neighbors, and route summarization was introduced to reduce the size of the routing table.

#### RIP-1

RIP version 1 (RIP-1) is a classful routing protocol, which supports only the broadcast of protocol packets. **Figure 1-198** shows the format of a RIP-1 packet. A RIP packet can carry a maximum of 25 routing entries. RIP is based on UDP, and a RIP-1 packet cannot be longer than 512 bytes. RIP-1 packets do not carry any mask information, and RIP-1 can identify only the routes to natural network segments, such as Class A, Class B, and Class C. Therefore, RIP-1 does not support route summarization or discontinuous subnets.

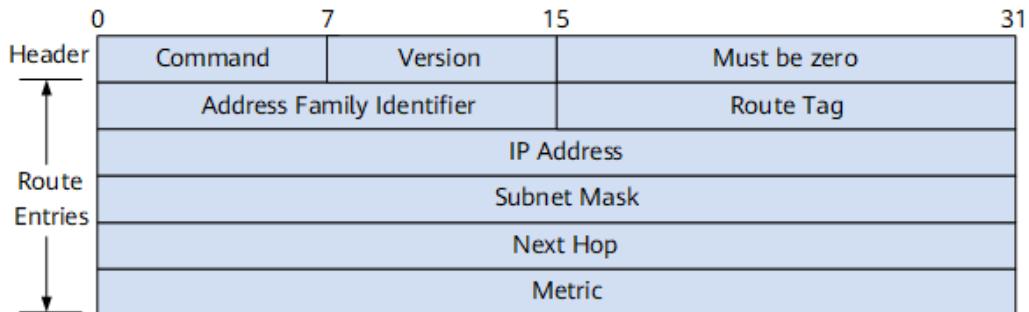
**Figure 1-198** RIP-1 packet format



## RIP-2

RIP version 2 (RIP-2) is a classless routing protocol. [Figure 1-199](#) shows the format of a RIP-2 packet.

**Figure 1-199** RIP-2 packet format



Compared with RIP-1, RIP-2 has the following advantages:

- Supports external route tags and uses a routing policy to flexibly control routes based on the tag.
- Supports route summarization and classless inter-domain routing (CIDR) by adding mask information to RIP-2 packets.
- Supports next hop specification so that the optimal next hop address can be specified on the broadcast network.
- Supports Update packets transmission along multicast routes. Only the routers that support RIP-2 can receive RIP-2 packets, which reduces resource consumption.
- Provides three packet authentication modes: simple text authentication, Message Digest 5 (MD5) authentication and HMAC-SHA256 authentication. For the sake of security, using the HMAC-SHA256 authentication is recommended.

## Timers

RIP uses the following four timers:

- Update timer: The Update timer periodically triggers Update packet transmission. By default, the interval at which Update packets are sent is 30s.
- Age timer: If a device does not receive any packets from its neighbor to update a route before the route expires, the device considers the route unreachable. The default value of the Age timer is 180s.
- Garbage-collect timer: If a route becomes invalid (the Age timer expires or a message indicating that the route is unreachable is received), the route is placed into a garbage queue instead of being immediately deleted from the RIP routing table. The garbage-collect timer is used by the device to monitor the garbage queue and delete expired routes. If a route's Update packet is received before the garbage-collect timer expires, the route is placed back into the age queue. The purpose of setting the garbage-collect timer is to prevent route flapping. The default value of garbage-collect timer is 120s.

- Hold-down timer: If a RIP device receives an updated route with cost 16 from a neighbor, the route enters the hold-down state, and the hold-down timer is started. To prevent route flapping, the RIP device accepts new route updates from the neighbor before the hold-down timer expires only when the following conditions are met:
  - a. The cost carried in the Update packet is less than or equal to that carried in the previous Update packet.
  - b. The hold-down timer expires, and the router enters the garbage-collect state.

The relationship between RIP routes and the four timers is as follows:

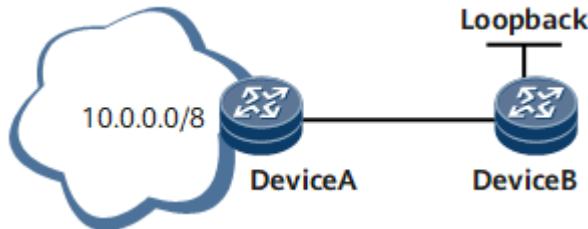
- The advertisement of RIP update information is controlled by the Update timer. By default, RIP update information is sent every 30 seconds.
- Each routing entry is associated with two timers: age timer and garbage-collect timer.
  - a. Whenever a route is learned and added to the routing table, the age timer is started.
  - b. If no Update packet is received from the neighbor within 180 seconds after the age timer is started, the metric of the corresponding route is set to 16, and the garbage-collect timer is started.
- If no Update packet is received within 120 seconds after the garbage-collect timer is started, the corresponding routing entry is deleted from the routing table after the garbage-collect timer expires.
- By default, the hold-down timer of RIP is disabled. If the hold-down timer is enabled manually, it starts after the system receives a route with a cost greater than 16 from a neighbor.

## Split Horizon

### Split Horizon on Broadcast, P2MP, and P2P Networks

Split horizon prevents a RIP-enabled interface from sending back the routes it learns, which reduces bandwidth consumption and prevents routing loops.

**Figure 1-200** Networking for interface-based split horizon



In **Figure 1-200**, Device A sends Device B a route to 10.0.0.0/8. If split horizon is not configured, Device B will send this route back to Device A after learning it from Device A. As a result, Device A learns the following routes to 10.0.0.0/8:

- A direct route with zero hops

- A route with Device B as the next hop and total two hops

Only direct routes, however, are active in the RIP routing table of Device A.

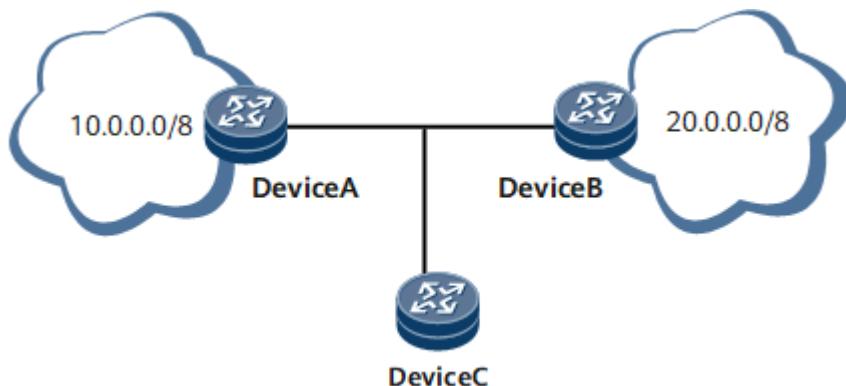
If the route from Device A to 10.0.0.0/8 becomes unreachable and Device B is not notified, Device B still considers the route to 10.0.0.0/8 reachable and continues sending this route to Device A. Then, Device A receives incorrect routing information and considers the route to 10.0.0.0/8 reachable through Device B; Device B considers the route to 10.0.0.0/8 reachable through Device A. As a result, a loop occurs on the network.

After split horizon is configured, Router B no longer sends the route back after learning the route, which prevents such a loop.

## Split Horizon on NBMA Networks

On a Non-Broadcast Multi-Access (NBMA) network where an interface is connected to multiple neighbors, RIP supports neighbor-based split horizon. On NBMA networks, routes are sent in unicast mode, and an interface can differentiate which neighbor each route was learned from, and the interface will not send the routes back to the neighbor it learned them from.

**Figure 1-201** Networking for neighbor-based split horizon on an NBMA network

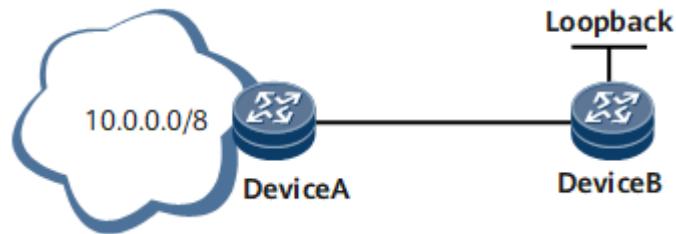


In **Figure 1-201**, Device A sends the route to 10.0.0.0/8 that it learns from Device B only to Device C.

## Poison Reverse

Poison reverse allows a RIP-enabled interface to set the cost of the route that it learns from a neighbor to 16 (indicating that the route is unreachable) and then send the route back. After receiving this route, the neighbor deletes the useless route from its routing table, which prevents loops.

Figure 1-202 Networking for poison reverse



In **Figure 1-202**, Device A sends Device B a route to 10.0.0.0/8. If poison reverse is not configured, Device B will send this route back to Device A after learning it from Device A. As a result, Device A learns the following routes to 10.0.0.0/8:

- A direct route with zero hops
- A route with Device B as the next hop and total two hops

Only direct routes, however, are active in the RIP routing table of Device A.

If the route from Device A to 10.0.0.0 becomes unreachable and Device B is not notified, Device B still considers the route to 10.0.0.0/8 reachable and continues sending this route to Device A. Then, Device A receives incorrect routing information and considers the route to 10.0.0.0/8 reachable through Device B; Device B considers the route to 10.0.0.0/8 reachable through Device A. As a result, a loop occurs on the network.

With poison reverse, after Device B receives the route from Device A, Device B sends a route unreachable message to Device A with cost 16. Device A then no longer learns the reachable route from Device B, which prevents routing loops.

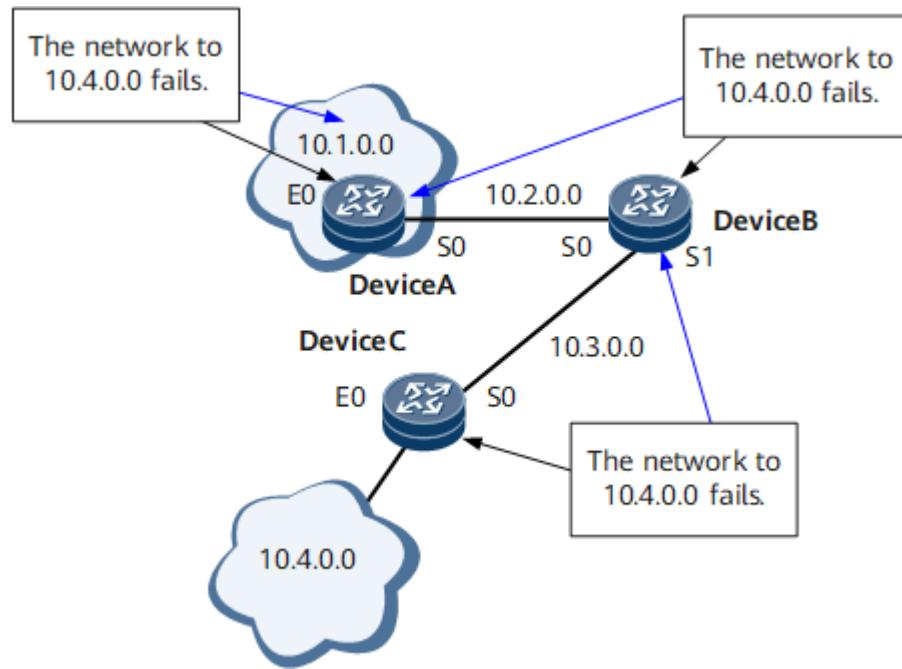
If both split horizon and poison reverse are configured, only poison reverse takes effect.

## Triggered Update

The principle of triggered update is that when the routing information changes, it immediately sends a triggered update message to its neighbor to notify the changed routing information.

Triggered update allows a device to advertise routing information changes immediately, which speeds up network convergence.

Figure 1-203 Networking for triggered update



In **Figure 1-203**, if the route to 10.4.0.0 becomes unreachable, Device C learns the information first. By default, a RIP-enabled device sends routing updates to its neighbors every 30s. If Device C receives an Update packet from Device B within 30s while Device C is still waiting to send Update packets, Device C learns the incorrect route to 10.4.0.0. In this case, the next hops of the routes from Device B or Device C to network 10.4.0.0 are Device C and Device B respectively, which results in routing loops. If Device C sends an Update packet to Device B immediately after it detects a network, Device B can rapidly update its routing table, which prevents routing loops.

In addition, if the next hop of a route becomes unavailable due to a link failure, the local Device sets the cost of the route to 16 and then advertises the route immediately to its neighbors. This process is called route poisoning.

## Route Summarization

Route summarization allows routes to the same natural network segment but different subnets to be summarized into a single route belonging to the same network segment before it is transmitted to other network segments. RIP-1 packets do not carry mask information, and therefore RIP-1 can advertise only routes with natural masks. RIP-2 supports route summarization because RIP-2 packets carry mask information. Therefore, RIP-2 supports subnetting.

In RIP-2, route summarization can reduce the size of the routing table and improve the extensibility and efficiency of a large-scale network.

Route summarization has two modes:

- Process-based classful summarization

Summary routes are advertised with natural masks. If split horizon or poison reverse is configured, classful summarization becomes invalid because split

horizon or poison reverse suppresses some routes from being advertised. In addition, when classful summarization is configured, routes learned from different interfaces may be summarized into a single route. As a result, a conflict occurs in the advertisement of the summary route.

For example, a RIP process summarizes the route 10.1.1.0 /24 with metric 2 and route 10.2.2.0/24 with metric 3 into the route 10.0.0.0/8 with metric 2.

- Interface-based summarization

Users can specify a summary address.

For example, users can configure a RIP-enabled interface to summarize the route 10.1.1.0/24 with metric 2 and route 10.1.2.0/24 with metric 3 into the route 10.1.0.0/16 with metric 2.

## Multi-Process and Multi-Instance

RIP supports multi-process and multi-instance to simplify network management and improve service control efficiency. Multi-process allows a set of interfaces to be associated with a specific RIP process, which ensures that the specific RIP process performs all the protocol operations only on this set of interfaces. Therefore, multiple RIP processes can run on one router, and each process manages a unique set of interfaces. In addition, the routing data of each RIP process is independent; however, processes can import routes from each other.

On routers that support VPN, each RIP process is associated with a specific VPN instance. Therefore, all the interfaces associated with the RIP process need to be associated with the RIP process-related VPN instance.

## BFD for RIP

### Background

Routing Information Protocol (RIP)-capable devices monitor the neighbor status by exchanging Update packets periodically. During the period local devices detect link failures, carriers or users may lose a large number of packets. Bidirectional forwarding detection (BFD) for RIP can speed up fault detection and route convergence, which improves network reliability.

After BFD for RIP is configured on the router, BFD can detect a fault (if any) within milliseconds and notify the RIP module of the fault. The router then deletes the route that passes through the faulty link and switches traffic to a backup link. This process speeds up RIP convergence.

**Table 1-79** describes the differences before and after BFD for RIP is configured.

**Table 1-79** Differences before and after BFD for RIP is configured

| Item                           | Link Fault Detection Mechanism | Convergence Speed |
|--------------------------------|--------------------------------|-------------------|
| BFD for RIP is not configured. | A RIP aging timer expires.     | Second-level      |

| Item                       | Link Fault Detection Mechanism | Convergence Speed |
|----------------------------|--------------------------------|-------------------|
| BFD for RIP is configured. | A BFD session goes Down.       | Millisecond-level |

## Related Concepts

The BFD mechanism bidirectionally monitors data protocol connectivity over the link between two routers. After BFD is associated with a routing protocol, BFD can rapidly detect a fault (if any) and notify the protocol module of the fault, which speeds up route convergence and minimizes traffic loss.

BFD is classified into the following modes:

- Static BFD

In static BFD mode, BFD session parameters (including local and remote discriminators) must be configured, and requests must be delivered manually to establish BFD sessions.

Static BFD is applicable to networks on which only a few links require high reliability.

- Dynamic BFD

In dynamic BFD mode, the establishment of BFD sessions is triggered by routing protocols, and the local discriminator is dynamically allocated, whereas the remote discriminator is obtained from BFD packets sent by the neighbor.

When a new neighbor relationship is set up, a BFD session is established based on the neighbor and detection parameters, including source and destination IP addresses. When a fault occurs on the link, the routing protocol associated with BFD can detect the BFD session Down event. Traffic is switched to the backup link immediately, which minimizes data loss.

Dynamic BFD is applicable to networks that require high reliability.

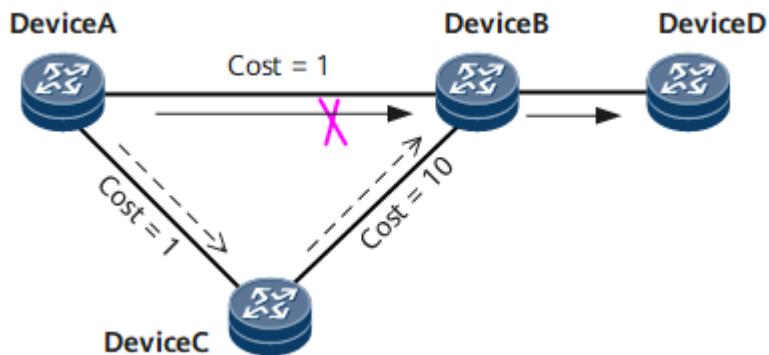
## Implementation

For details about BFD implementation, see "BFD" in *Router Feature Description - Reliability*. [Figure 1-204](#) shows a typical network topology for BFD for RIP.

- Dynamic BFD for RIP implementation:
  - a. RIP neighbor relationships are established among Device A, Device B, and Device C and between Device B and Device D.
  - b. BFD for RIP is enabled on Device A and Device B.
  - c. Device A calculates routes, and the next hop along the route from Device A to Device D is Device B.
  - d. If a fault occurs on the link between Device A and Device B, BFD will rapidly detect the fault and report it to Device A. Device A then deletes the route whose next hop is Device B from the routing table.
  - e. Device A recalculates routes and selects a new path Device C → Device B → Device D.

- f. After the link between Device A and Device B recovers, a new BFD session is established between the two routers. Device A then reselects an optimal link to forward packets.
- Static BFD for RIP implementation:
  - a. RIP neighbor relationships are established among Device A, Device B, and Device C and between Device B and Device D.
  - b. Static BFD is configured on the interface that connects Device A to Device B.
  - c. If a fault occurs on the link between Device A and Device B, BFD will rapidly detect the fault and report it to Device A. Device A then deletes the route whose next hop is Device B from the routing table.
  - d. After the link between Device A and Device B recovers, a new BFD session is established between the two routers. Device A then reselects an optimal link to forward packets.

**Figure 1-204 BFD for RIP**



## Usage Scenario

BFD for RIP is applicable to networks that require high reliability.

## Benefits

BFD for RIP improves network reliability and enables devices to rapidly detect link faults, which speeds up route convergence on RIP networks.

## RIP NSR

Devices with a distributed architecture support RIP Non-stop Routing (NSR). RIP backs up all route data from the Active Main Board (AMB) to the Standby Main Board (SMB). Whenever the AMB fails, the SMB becomes active and takes over traffic. RIP NSR ensures that routes are synchronous between the AMB and SMB. Therefore, during the AMB/SMB switchover, the neighbor will not detect the fault on the local device.

## RIP Authentication

As networks develop, there has been considerable growth in all types of data, voice, and video information exchanged on networks. In addition, new services, such as E-commerce, online conferencing and auctions, video on demand (VoD),

and e-learning have sprung up increasingly, requiring higher information security than before. Carriers must protect data packets from being illegally obtained or modified by attackers and prohibit unauthorized users from accessing network resources. RIP packet authentication effectively meets these security requirements.

RIP authentication falls into the following modes:

- Simple authentication: The authenticated party adds the configured password directly to packets for authentication. This authentication mode provides the lowest password security.
- MD5 authentication: The authenticated party uses the Message Digest 5 (MD5) algorithm to generate a ciphertext password and adds it to packets for authentication. This authentication mode improves password security. For the sake of security, using the HMAC-SHA256 algorithm rather than the MD5 algorithm is recommended.
- Keychain authentication: The authenticated party configures a keychain that changes over time. This authentication mode further improves password security.

Keychain authentication improves RIP security by periodically changing the password and the encryption algorithms. For details about Keychain, see "Keychain" in NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X Feature Description - Security.

- HMAC-SHA256 authentication: The authenticated party uses the HMAC-SHA256 algorithm to generate a ciphertext password and adds it to packets for authentication.

RIP authentication ensures network security by adding an authentication field used to encrypt a packet before sending the packet to ensure network security. After receiving a RIP packet from a remote router, the local router discards the packet if the authentication password in the packet does not match the local authentication password. This authentication mode protects the local router.

On IP networks of carriers, RIP authentication ensures the secure transmission of packets, improves the system security, and provides secure network services for carriers.

### 1.1.6.2 RIP Configuration

RIP can advertise and receive routes to select routes for data forwarding and provide the Network Management Station (NMS) function. RIP is applicable to small-scale networks.

#### 1.1.6.2.1 Overview of RIP

RIP protocol can implement the interworking of small and medium-sized networks.

#### Definition

Routing Information Protocol (RIP) is a simple Interior Gateway Protocol (IGP). RIP is used in small-scale networks, such as campus networks and simple regional networks.

As a distance-vector routing protocol, RIP exchanges routing information through User Datagram Protocol (UDP) packets with port number 520.

RIP employs the hop count as the metric to measure the distance to the destination. In RIP, by default, the number of hops from the router to its directly connected network is 0; the number of hops from the router to a network that is reachable through another router is 1, and so on. The hop count (the metric) equals the number of routers along the path from the local network to the destination network. To speed up route convergence, RIP defines the hop count as an integer that ranges from 0 to 15. A hop count that is greater than or equal to 16 is classified as infinite, indicating that the destination network or host is unreachable. Due to the hop limit, RIP is not applicable to large-scale networks.

RIP has two versions:

- RIP version 1 (RIP-1), a classful routing protocol
- RIP version 2 (RIP-2), a classless routing protocol

RIP supports split horizon, poison reverse, and triggered update, which improves the performance and prevents routing loops.

## Purpose

As the earliest IGP, RIP is used in small and medium-sized networks. Its implementation is simple, and the configuration and maintenance of RIP are easier than those of Open Shortest Path First (OSPF) and Intermediate System-to-Intermediate System (IS-IS). Therefore, RIP is widely used on live networks.

### 1.1.6.2.2 Configuration Precautions for RIP

## Feature Requirements

**Table 1-80** Feature requirements

| Feature Requirements                                                                                                                                                                                                                                                                                                                                 | Series           | Models                                                                                     |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| After the device is restarted, if the BFD session status of the local device or neighbor is AdminDown, the RIP status is not affected. When the BFD session is renegotiated, if BFD reports the detection status Down but the previous detection status is Up, RIP sets the neighbor status to Down. In other cases, the RIP status is not affected. | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

### 1.1.6.2.3 Configuring Basic RIP Functions

To use RIP features, start RIP and specify the network segment and RIP version first.

## Usage Scenario

Configuring basic RIP functions is a prerequisite for building RIP networks.

## Pre-configuration Tasks

Before configuring basic RIP functions, complete the following tasks:

- Configure the link layer protocol.
- Configure network layer addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.

## Creating a RIP process

The creation of a RIP process is a precondition of all RIP configurations.

## Usage Scenario

Before running RIP on a device, you need to create a RIP process on the device.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

RIP supports multi-instance. To bind a RIP process to a VPN instance, you can run the **rip [ process-id ] vpn-instance vpn-instance-name** command.

### Step 3 (Optional) Run **description description**

A description is configured for the RIP process.

### Step 4 Run **commit**

The configuration is committed.

----End

## Follow-up Procedure

If you run RIP-related commands in the interface view before enabling RIP, the configurations take effect only after RIP is enabled.

## Enabling RIP on Specified Network Segments

To enable a RIP process to receive and send RIP routes on specified network segments, enable RIP for the network segments first.

## Context

You can perform either of the following operations to enable RIP on specified network segments:

- Run the **network** command in the RIP process view to enable a RIP process to receive and send routes on a specified network segment.

- Run the **rip enable** command in the interface view to enable a RIP process to receive and send routes on all the network segments where a specified interface is located.

 **NOTE**

The **rip enable** command takes precedence over the **network** command.

## Procedure

- Enable a RIP process to receive and send routes on a specified network segment.
  - Run **system-view**  
The system view is displayed.
  - Run **rip [ process-id ]**  
A RIP process is created, and the view of the RIP process is displayed.
  - Run **network network-address**  
RIP is enabled on a specified network segment.
  - Run **commit**  
The configuration is committed.  
RIP runs only on interfaces on the specified network segment.
- Enable a RIP process to receive and send routes on all specified network segments where a specified interface is located.
  - Run **system-view**  
The system view is displayed.
  - Run **interface interface-type interface-number**  
The interface view is displayed.
  - Run **rip enable process-id**  
RIP is enabled on all the network segments where the interface is located.
  - Run **commit**  
The configuration is committed.

----End

## (Optional) Configuring the RIP Version Number

RIP include RIP-1 and RIP-2. The two versions have different functions.

## Procedure

- Configure the global RIP version number.
  - Run **system-view**  
The system view is displayed.
  - Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

- c. Run **version** *version-num*

The global RIP version number is specified.

 NOTE

The RIP-1 protocol poses a security risk, and therefore the RIP-2 protocol is recommended.

- d. Run **commit**

The configuration is committed.

- Configure the RIP version number on an interface.

- a. Run **system-view**

The system view is displayed.

- b. Run **interface** *interface-type interface-number*

The interface view is displayed.

- c. Run **rip version** { 1 | 2 [ **broadcast** | **multicast** ] }

The RIP version number is specified for the interface.

 NOTE

The RIP-1 protocol poses a security risk, and therefore the RIP-2 protocol is recommended.

- d. Run **commit**

The configuration is committed.

----End

## (Optional) Configuring a RIP Preference

When there are routes discovered by multiple routing protocols on the same device, you can set a preference for RIP to control the route selection result.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip** [ *process-id* ]

A RIP process is created, and the RIP view is displayed.

#### Step 3 Run **preference** { *preference* | { **route-policy** *route-policy-name* | **route-filter** *route-filter-name* } } \*

A preference is set for RIP.

The **preference** command can be used together with a route-policy to set a preference for the matched routes.

If the RIP preference is changed after RIP routing information is delivered to the routing management (RM) module, the RM module re-updates the routing table.

#### Step 4 Run commit

The configuration is committed.

----End

### (Optional) Disabling Check on the Source Address of RIP Packets on a P2P Network

On a P2P network, IP addresses of interfaces at the two ends of a link may belong to different networks. Therefore, the two ends can receive packets from each other only when the check on the source addresses of RIP packets is disabled.

#### Context

By default, RIP checks the source addresses of received packets, and the local RIP interface receives only the packets from the same network. On a P2P network, IP addresses of interfaces at the two ends of a link may belong to different networks. Therefore, the two ends can receive packets from each other only when the check on the source addresses of RIP packets is disabled.

#### Procedure

##### Step 1 Run system-view

The system view is displayed.

##### Step 2 Run rip [*process-id*]

A RIP process is created, and the RIP view is displayed.

##### Step 3 Run undo verify-source

The check on the source address of RIP packets is disabled.

##### Step 4 Run commit

The configuration is committed.

----End

### (Optional) Enabling Check on Zero Fields of RIP-1 Packets

Certain fields in RIP-1 packets must be 0, and these fields are called zero fields.

#### Context

Certain fields in RIP-1 packets must be 0, and RIP-2 packets contain no zero fields.

#### Procedure

##### Step 1 Run system-view

The system view is displayed.

##### Step 2 Run rip [*process-id*]

A RIP process is created, and the RIP view is displayed.

### Step 3 Run **checkzero**

The check on zero fields of RIP-1 packets is configured.

### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring an NBMA Network

RIP packets on a Non-Broadcast Multiple Access (NBMA) network are sent in a mode different from those on networks of other types, and therefore, special configurations are required.

## Usage Scenario

RIP packets are sent in unicast mode only on an NBMA network. On networks of other types, RIP packets are sent in either broadcast or multicast mode.

Therefore, you need to perform the following configurations for an NBMA network:

- Specify RIP neighbors.
- Prevent interfaces from sending RIP packets in either broadcast or multicast mode.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

### Step 3 Run **peer peer-address**

RIP neighbors are configured on the NBMA network.

### Step 4 Perform either of the following operations as required:

- Run the **silent-interface all** command to prevent all interfaces from sending RIP packets in either broadcast or multicast mode.
- Run the **silent-interface interface-type interface-number** command to prevent a specified interface from sending RIP packets in either broadcast or multicast mode.

#### NOTE

If you want a small number of interfaces to send RIP packets in either broadcast or multicast mode, you can run the **silent-interface all** command first to prevent all interfaces from sending RIP packets in either broadcast or multicast mode and then run the **silent-interface disable interface-type interface-number** command to restore the capability to send RIP packets in either broadcast or multicast mode for the small number of interfaces.

### Step 5 Run commit

The configuration is committed.

----End

## Verifying the Configuration

After basic RIP functions are configured, you can view the current running status of RIP and RIP routing information.

## Prerequisites

All the basic RIP functions have been configured.

## Procedure

- Run the **display rip [ process-id | vpn-instance vpn-instance-name ]** command to check RIP running status and configuration.
- Run the **display rip process-id route** command to check RIP routes.

----End

### 1.1.6.2.4 Preventing Routing Loops

RIP is based on the DV algorithm. RIP devices advertise their routing tables to their neighbors, and therefore, routing loops may occur.

## Usage Scenario

RIP prevents routing loops through the following mechanisms:

- Counting to infinity: RIP defines the cost 16 as infinity. If the cost of a route reaches 16 due to a routing loop, this route is considered unreachable.
- Split horizon: Split horizon prevents a RIP-enabled interface from sending back the routes it learns, which reduces bandwidth consumption and prevents routing loops.
- Poison reverse: Poison reverse allows a RIP-enabled interface to set the cost of the route that it learns from a neighbor to 16 (indicating that the route is unreachable) and then send the route back. After receiving this route, the neighbor deletes the useless route from its routing table, which prevents loops.
- Suppression timers: Suppression timers can prevent routing loops and reduce the possibility of resulting in incorrect routing information due to the receiving of incorrect routes.
- Disabling an interface from receiving and sending RIP packets: Similar to split horizon or poison reverse, this function filters out unreliable IP routes. However, routing information on the network may be incorrect because neighbors cannot receive packets from the local router.

#### NOTE

Counting to infinity is a basic feature of RIP, and therefore, it does not need to be configured. Split horizon and poison reverse, however, need to be configured. When both split horizon and poison reverse are configured, only poison reverse takes effect.

## Pre-configuration Tasks

Before configuring RIP to prevent routing loops on the network, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.6.2.3 Configuring Basic RIP Functions](#)

## Configuration Procedure

Perform one or more of the following configurations as required.

### Configuring Split Horizon

You can configure split horizon to prevent routing loops.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

##### Step 3 Run **rip split-horizon**

Split horizon is enabled.

##### Step 4 Run **commit**

The configuration is committed.

----End

#### Follow-up Procedure

If both split horizon and poison reverse are configured, only poison reverse takes effect.

### Configuring Poison Reverse

You can configure poison reverse to prevent routing loops.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

##### Step 3 Run **rip poison-reverse**

Poison reverse is enabled.

#### Step 4 Run commit

The configuration is committed.

----End

### Follow-up Procedure

If both split horizon and poison reverse are configured, only poison reverse takes effect.

### Configuring Suppression Timers

Suppression timers can prevent routing loops and reduce the possibility of generating incorrect routing information due to the receiving of incorrect routes.

### Context

When hop count of a route increases, a device starts suppression timers and accepts the Update packet of this route and updates the routing table until the suppression timers expire.

Suppression timers delays the addition of incorrect routes to the routing table and slows down route convergence on the entire network as well. Therefore, exercise caution when configuring the suppression timers.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run rip process-id

A RIP process is created, and the RIP view is displayed.

#### Step 3 Run timers rip update age suppress garbage-collect

The suppression timers are set.

#### Step 4 Run commit

The configuration is committed.

----End

### Follow-up Procedure

RIP has four timers: *update*, *age*, *suppress*, and *garbage-collect*. The value of *update* is less than that of *age*, and the value of *suppress* is less than that of *garbage-collect*. Setting improper values for the timers affects RIP convergence speed and even causes route flapping on the network. For example, if the value of *update* is greater than that of *age*, a device cannot inform its neighbors of the change of RIP routes immediately.

For the configurations of *update*, *age*, *suppress*, and *garbage-collect*, see [Configuring RIP Timers](#).

## Verifying the Configuration

After routing loop prevention is configured, you can view the current running status of RIP, information about interfaces, and RIP routing information.

### Prerequisites

Routing loop prevention has been configured.

### Procedure

- Run the **display rip [ process-id | vpn-instance vpn-instance-name ]** command to check RIP running status and configuration.
- Run the **display rip process-id route** command to check RIP routes.
- Run the **display rip process-id interface** command to check information about RIP interfaces.

----End

### 1.1.6.2.5 Adjusting RIP Route Selection

You can adjust RIP route selection on a complicated network.

### Usage Scenario

The implementation of RIP is simple, and therefore, RIP is widely used in small and medium networks. To flexibly apply RIP on the live network to meet various requirements of users, you can change RIP route selection by setting different parameters.

### Pre-configuration Tasks

Before adjusting RIP route selection, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.6.2.3 Configuring Basic RIP Functions](#).

### Configuration Procedure

Perform one or more of the following configurations as required.

### Disabling RIP-2 Classful Summarization

On the network where subnets are incontiguous, you can cancel the classful summarization of RIP-2 so that more accurate routing information can be obtained.

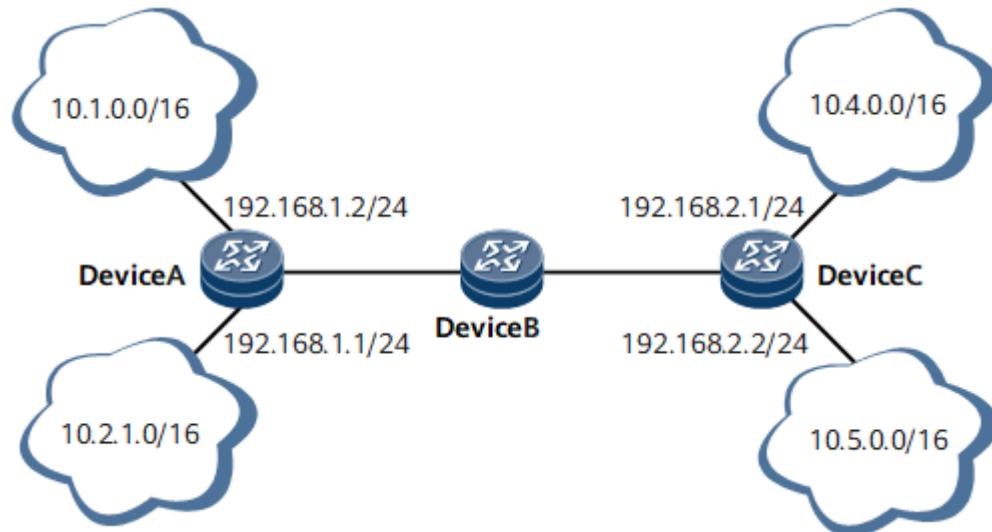
### Context

Summarizing IP addresses can reduce the routing table size, but it shields routing information of subnets. As a result, incorrect routing information may be calculated.

On a non-contiguous subnet, you need to disable RIP-2 classful summarization. On the network shown in [Figure 1-205](#), you need to disable split horizon from interfaces of Router A and Router C.

By default, RIP-2 classful summarization is enabled. Therefore, Router B and Router C send a route destined for 10.0.0.0/8 to Router A. Router A cannot differentiate 10.1.0.0/16, 10.2.0.0/16, 10.4.0.0/16, or 10.5.0.0/16. As a result, incorrect routes are calculated.

**Figure 1-205** Disabling RIP-2 classful summarization



## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

**Step 3** Run **undo summary**

RIP-2 classful summarization is disabled.

**Step 4** Run **commit**

The configuration is committed.

----End

## Configuring the Additional Metric on an Interface

The additional metric is a metric (number of hops) that is added to the original metric of an RIP route. You set additional metrics for received RIP routes and those to be sent.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Set the metric to be added to received routes.

Run any of the following commands as required:

- Based on the basic ACL:

a. Run **rip metricin { value | { acl-number| acl-name acl-name } value1 }**

The metric to be added to the received routes that match the specified basic ACL is configured.

b. Run **quit**

Return to the system view.

c. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The ACL view is displayed.

d. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the

unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- Based on an IP prefix list:

Run **rip metricin { value | ip-prefix ip-prefix-name value1 }**

The metric to be added to the received routes that match the specified basic ACL is configured.

#### Step 4 Configure a metric to be added to the routes to be advertised by the interface.

Run any of the following commands as required:

- Based on the basic ACL:

- Run **rip metricout { value | { acl-number | acl-name acl-name } value1 }**  
\*

The metric to be added to the routes that match the specified basic ACL and are to be advertised is configured.

- Run **quit**

Return to the system view.

- Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The ACL view is displayed.

- Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule is configured for the ACL.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.

- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:
  - Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.
  - Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.
- Based on an IP prefix list:

Run **rip metricout { value | ip-prefix ip-prefix-name value1 } \***

The metric to be added to the routes that match the specified IP prefix list and are to be advertised is configured.

#### NOTE

If an ACL or IP prefix list is specified in the **rip metricout** command to filter the received RIP routes to be advertised, the configured metric is added only to the matched routes. If a RIP route does not match the ACL or the IP prefix list, its metric is increased by 1. When an ACL or an IP prefix list is used with the **rip metricout** command, the additional metric ranges from 2 to 15.

#### Step 5 Run **commit**

The configuration is committed.

----End

### Setting the Maximum Number of Equal-Cost Routes

You can set the maximum number of equal-cost RIP routes to adjust the number of routes that load-balance traffic.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

#### Step 3 Run **maximum load-balancing number**

The maximum number of equal-cost routes is set.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After adjusting RIP route selection, you can view the current running status of RIP, information about interfaces, and RIP routing information.

## Prerequisites

The configurations of adjusting RIP route selection have been performed.

## Procedure

- Run the **display rip [ process-id | vpn-instance vpn-instance-name ]** command to check RIP running status and configuration.
- Run the **display rip process-id route** command to check RIP routes.
- Run the **display rip process-id interface** command to check information about RIP interfaces.

----End

### 1.1.6.2.6 Controlling RIP Routing Information

In most cases, multiple protocols run on the same network. Therefore, you need to control routing information of every protocol to meet different networking requirements.

## Usage Scenario

To meet the requirements of complex networking, accurately control the sending and receiving of RIP routing information.

## Pre-configuration Tasks

Before configuring a router to control RIP routing information, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer
- [1.1.6.2.3 Configuring Basic RIP Functions](#)

## Configuration Procedure

Perform one or more of the following configurations as required.

### Configuring RIP to Import External Routes

A RIP process can import the routes learned by other processes or routing protocols to enrich routing entries.

## Context

On a large-scale network, different routing protocols may be configured for devices in different areas. To implement communication between a RIP area and other routing areas, you need to configure the device to import routes from other routing protocols.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run rip [ *process-id* ]

A RIP process is created, and the RIP view is displayed.

### Step 3 (Optional) Run default-cost *cost*

The default metric is configured for imported routes.

If no metric is specified for imported external routes, the default metric 0 is used.

### Step 4 Run import-route

The device is configured to import external routes.

- To import direct routes, static routes, IS-IS routes, OSPF routes, or routes of another RIP process, run the **import-route { static | direct | bgp | unr | { isis | ospf | rip } [ *process-id* ] } [ cost *cost* | { route-policy *route-policy-name* | route-filter *route-filter-name* } ]** \* command.
- To import IBGP routes, run the **import-route bgp permit-ibgp [ cost *cost* | { route-policy *route-policy-name* | route-filter *route-filter-name* } ]** \* or **import-route bgp [ permit-ibgp ] { cost transparent { route-policy *route-policy-name* | route-filter *route-filter-name* } | { route-policy *route-policy-name* | route-filter *route-filter-name* } cost transparent | cost transparent }** command.

#### NOTE

Importing routes from other protocols into a RIP process may lead to routing loops. Therefore, exercise caution when running the **import-route** command.

### Step 5 Run commit

The configuration is committed.

----End

## Configuring RIP to Advertise Default Routes

Default routes are destined for 0.0.0.0.

## Context

In a routing table, the default route is a route to 0.0.0.0 with mask 0.0.0.0. You can run the **display ip routing-table** command to check whether the default route is configured. When the destination address of a packet does not match any entry in the routing table, the router forward this packet along a default route.

If the default route and the destination address of the packet do not exist in the routing table, the router discards the packet and sends an Internet Control Message Protocol (ICMP) packet, informing the originating host that the destination address or network is unreachable.

After the **default-route originate** command is run, default routes are advertised to RIP neighbors only when there are default routes in the routing table, and the default route that is learned from a neighbor is deleted.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run rip [ process-id ]

A RIP process is created, and the RIP view is displayed.

### Step 3 Run default-route originate [ cost cost | tag tag | { match default | { route-policy route-policy-name [ advertise-tag ] | route-filter route-filter-name } } [ avoid-learning ] ] \*

RIP is enabled to advertise default routes.

### Step 4 Run commit

The configuration is committed.

----End

## Configuring RIP to Filter Received Routes

You can configure an import policy for a device so that the device receives only required routes.

## Context

You can configure an import policy by specifying an ACL or IP prefix list to filter received routes. You can also configure a device to receive only the RIP packets from a specified neighbor.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run rip [ process-id ]

A RIP process is created, and the RIP view is displayed.

### Step 3 Configure RIP to filter received routes.

- Run **filter-policy acl-number import** [ *interface-type interface-number* ]

The learned routes are filtered based on the ACL.

- Run **filter-policy gateway ip-prefix-name import**

The routes advertised by neighbors are filtered based on the IP prefix list.

- Run **filter-policy acl-name acl-name import** [ *interface-type interface-number* ]

The learned routes are filtered based on the named ACL.

- Run **filter-policy ip-prefix ip-prefix-name [ gateway ip-prefix-name ] import [ interface-type interface-number ]**  
The routes learned from the specified interface are filtered based on the IP prefix list or interface.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring RIP to Filter the Routes to Be Advertised

You can set conditions to filter the routes to be advertised. Only the routes that meet the conditions can be advertised.

### Context

Devices can filter the routing information. To filter the routes to be advertised, you can configure an export filtering policy by specifying the ACL and IP prefix list.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run rip [ process-id ]

A RIP process is created, and the RIP view is displayed.

#### Step 3 Set the conditions to filter the advertised routes.

Run any of the following commands as required:

- Based on the basic ACL:
  - Run **filter-policy { acl-number | acl-name acl-name } export [ bgp | direct | static | unr | { isis | ospf | rip } [ process-id ] | interface-type interface-number ]**
  - Run **quit**  
Return to the system view.
  - Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**  
The ACL view is displayed.
  - Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***  
A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- Based on the IP prefix:

```
Run filter-policy ip-prefix ip-prefix-name export [bgp | direct | static | unr | { isis | ospf | rip } [process-id] | interface-type interface-number]
```

#### Step 4 Run commit

The configuration is committed.

----End

### Disabling RIP from Receiving Host Routes

You can disable RIP from receiving host routes on a device to prevent the device from receiving a large number of unwanted routes. Such configuration can reduce network resource consumption.

### Context

In some situations, the router may receive a large number of host routes from the same network segment. These routes, useless for routing, consume many network resources. By disabling RIP from receiving host routes, you can configure the router to reject the received host routes.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

### Step 3 Run **undo host-route**

RIP is disabled from receiving host routes.

### Step 4 Run **commit**

The configuration is committed.

----End

## Disabling the Check on RIP Packets with Cost 0

To communicate with other devices supporting packets with cost 0, prevent the Huawei device from checking the RIP packets with cost 0.

## Context

On the live network, not all devices can receive the packets with cost 0. By default, devices do not accept the packets with cost 0. Therefore, RIP interfaces discard the RIP packets with cost 0. To implement communication between Huawei devices and the devices that can accept the packets with cost 0, run the **undo zero-metric-check** command to allow interfaces to accept the RIP packets with cost 0.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

### Step 3 Run **undo zero-metric-check**

The interface is allowed to receive RIP packets with cost 0.

### Step 4 Run **commit**

The configuration is committed.

----End

## Follow-up Procedure

To enable the check on the RIP packets with cost 0, run the **zero-metric-check** command.

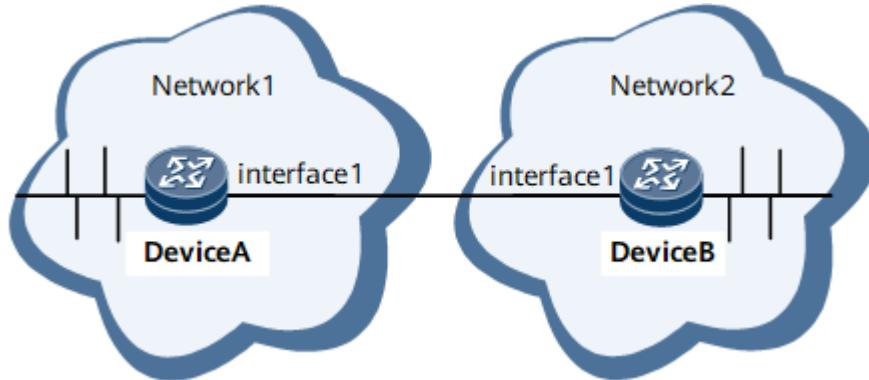
## Disabling an Interface from Sending Packets

When you do not need an interface connected to an external network to send routing information, disable the interface from sending RIP packets.

### Usage Scenario

On the network shown in [Figure 1-206](#), RIP-enabled Network 1 is connected to Network 2 through the edge device Device A. You can disable the interface on Device A from sending packets.

**Figure 1-206** Scenario where an interface is disabled from sending packets



### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 3** Run **undo rip output**

The interface is disabled from sending RIP packets.

**Step 4** Run **commit**

The configuration is committed.

----End

## Disabling an Interface from Receiving Packets

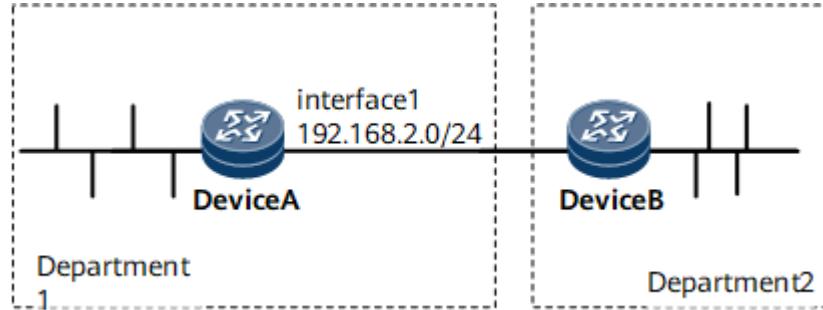
If an interface does not need to learn routing information from neighbors, you can disable the interface from receiving RIP packets.

### Usage Scenario

On an enterprise network where departments are not allowed to communicate with each other, you can disable interfaces from receiving packets.

On the network shown in [Figure 1-207](#), if you do not want Department 1 to learn routing information about Department 2, you can disable the interface on Device A from receiving packets.

**Figure 1-207** Scenario where an interface is disabled from receiving packets



## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 3** Run **undo rip input**

The interface is disabled from receiving RIP packets.

**Step 4** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After RIP routing information is successfully controlled, you can view all active routes in the RIP database and the current running status.

## Prerequisites

The configurations for controlling RIP routing information have been performed.

## Procedure

- Run the **display rip process-id database** command to check all active routes in the RIP database.
- Run the **display rip process-id** command to check the running status and configurations of the specified RIP process.

----End

### 1.1.6.2.7 Configuring RIP Fast Convergence

The network convergence speed is one of the key factors used to evaluate network performance.

#### Usage Scenario

The route convergence speed on a device is a performance index used to measure the network quality. Fast route convergence can improve the accuracy of routing information on the network.

#### Pre-configuration Tasks

Before configuring RIP fast convergence, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.6.2.3 Configuring Basic RIP Functions](#)

#### Configuration Procedure

Perform one or more of the following configurations as required.

##### Configuring RIP-2 Route Summarization

Configuring RIP-2 route summarization reduces the routing table size, which improves system performance and network security.

#### Context

On a medium-or large-sized network, the routing table contains a large number of routes. Storing and managing the routes consume a large number of memory resources. To resolve this problem, RIP-2 provides the route summarization function. Route summarization allows multiple routes with the same IP prefix and destination IP address to be summarized into one, which reduces the number of routes in the routing table and minimizes system resource consumption. In addition, if a specific link frequently alternates between Up and Down, the links not involved in the route summarization will not be affected, which prevents route flapping and improves network stability.

##### NOTE

RIP-1 does not support route summarization.

#### Procedure

- Enable RIP-2 automatic route summarization.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **rip [ process-id ]**  
A RIP process is created, and the RIP view is displayed.
  - c. Run **summary [ always ]**

RIP-2 automatic route summarization is enabled.

 NOTE

If you specify **always**, RIP-2 automatic route summarization takes effect, regardless of whether split horizon and poison reverse has been configured.

d. Run **commit**

The configuration is committed.

- Configure RIP-2 to advertise the summarized route.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **rip summary-address ip-address mask [ avoid-feedback ]**

RIP-2 is configured to advertise the summarized route to the specified IP address.

If you specify **avoid-feedback**, the local interface does not learn the summarized route whose destination address is the same as that of the advertised summarized route, which prevents routing loops.

d. Run **commit**

The configuration is committed.

----End

## Configuring RIP Timers

There are four RIP timers: Update, Age, Suppress, and Garbage-collect timers. You can adjust the RIP convergence speed by changing the values of RIP timers.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

**Step 3** Run **timers rip update age suppress garbage-collect**

RIP timers are configured.

**Step 4** Run **commit**

The configuration is committed.

----End

## Follow-up Procedure

The value of *update* is less than that of *age*, and the value of *suppress* is less than that of *garbage-collect*. Setting improper values for the timers affects RIP

convergence speed and even causes route flapping on the network. For example, if the value of *update* is greater than that of *age*, a device cannot inform its neighbors of the change of RIP routes immediately.

Configuring the Suppress timer can prevent routing loops. For details, see [Configuring Suppression Timers](#).

## Configuring RIP Triggered Update

You can speed up network convergence by changing the values of triggered update timers.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

#### Step 3 Run **timers rip triggered { minimum-interval *minimum-interval* | incremental-interval *incremental-interval* | maximum-interval *maximum-interval* } \***

RIP triggered update timers are configured.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Setting the Interval at Which Packets Are Sent and the Maximum Number of the Sent Packets

You can set the interval at which RIP packets are sent and the maximum number of packets that can be sent at a time to control the memory used by a device to process RIP update packets.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface *interface-type interface-number***

The interface view is displayed.

#### Step 3 Run **rip pkt-transmit { interval *interval* | number *pkt-count* | bandwidth *bandwidth-value* } \***

The interval at which RIP packets are sent and the maximum number of packets sent each time are set by the interface.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Setting the Maximum Length of RIP packets

You can increase the maximum length of RIP packets to enable them to carry more routes, which improves bandwidth utilization.

### Context

#### NOTICE

You can run the **rip max-packet-length** command to set a length greater than 512 bits for RIP packets only when the remote end can accept RIP packets longer than 512 bits.

After the maximum length of RIP packets is increased, Huawei devices may fail to communicate with non-Huawei devices. Therefore, exercise caution when running this command.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 3 Run **rip max-packet-length { value | mtu }**

The maximum length of RIP packets is set.

**mtu** specifies the maximum length of a RIP packet that can be accepted.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Setting the Maximum Number of RIP Routes

You can set the maximum number of RIP routes to make full use of network resources and improve network performance.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

**Step 3** Run **maximum-routes max-number [ threshold threshold-value ]**

The maximum number of routes is set.

**Step 4** Run **commit**

The configuration is committed.

----End

## Configuring RIP Hardware Replication

After RIP hardware replication is configured, RIP packets can be forwarded in hardware replication mode. This implements fast forwarding and optimizes system performance.

### Context

If a device has dot1q VLAN tag termination sub-interfaces, QinQ VLAN tag termination sub-interfaces, and user VLAN sub-interfaces and these sub-interfaces are configured with a large number of VLANs, protocol packets to be forwarded need to be replicated and broadcast in these VLANs. This increases the CPU load and may even cause problems such as packet replication and sending failures. To solve the problems, enable hardware replication for RIPv2 packets. After this function is enabled, hardware replicates the protocol packets to be broadcast, which reduces CPU consumption.

### Procedure

- Enable hardware replication on a dot1q VLAN tag termination sub-interface.
  - a. Run the **system-view** command to enter the system view.
  - b. Run the **interface interface-type interface-number . subinterface-number** command to enter the sub-interface view.
  - c. Run the **encapsulation dot1q-termination** command to configure the sub-interface as a dot1q VLAN tag termination sub-interface.
  - d. Run the **rip-broadcast-copy fast enable** command to enable hardware replication for RIPv2 packets.
  - e. Run the **commit** command to commit the configuration.
- Enable hardware replication on a QinQ VLAN tag termination sub-interface.
  - a. Run the **system-view** command to enter the system view.
  - b. Run the **interface interface-type interface-number . subinterface-number** command to enter the sub-interface view.
  - c. Run the **encapsulation qinq-termination** command to configure the sub-interface as a QinQ VLAN tag termination sub-interface.
  - d. Run the **rip-broadcast-copy fast enable** command to enable hardware replication for RIPv2 packets.
  - e. Run the **commit** command to commit the configuration.
- Enable hardware replication on a user VLAN sub-interface.
  - a. Run the **system-view** command to enter the system view.
  - b. Run the **interface interface-type interface-number . subinterface-number** command to enter the sub-interface view.

- c. Run the **user-vlan { start-vlan-id [ end-vlan-id ] | cevlan } qinq { start-pe-vlan [ end-pe-vlan ] | pevlan }** command to configure the sub-interface as a user VLAN sub-interface.
- d. Run the **rip-broadcast-copy fast enable** command to enable hardware replication for RIPv2 packets.
- e. Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

After configuring fast RIP convergence, check the running status of RIP, RIP routing information, all active routes in the RIP database, and information about interfaces.

## Prerequisites

Fast RIP convergence has been configured.

## Procedure

- Run the **display rip [ process-id | vpn-instance vpn-instance-name ]** command to check RIP running status and configuration.
- Run the **display rip process-id route** command to check RIP routes.
- Run the **display rip process-id database [ verbose ]** command to check all active routes in the RIP database.
- Run the **display rip process-id interface [ interface-type interface-number ] [ verbose ]** command to check information about RIP interfaces.

----End

### 1.1.6.2.8 Configuring Dynamic BFD for RIP

On a network that runs high-rate data services, dynamic BFD for RIP can be configured to quickly detect and respond to network faults.

## Usage Scenario

RIP-capable devices detect neighbor status by exchanging Update packets periodically. The default value of the aging timer is 180s. If a link fault occurs, RIP can detect the fault only after 180s. During this period, carriers or users may lose a large number of packets.

BFD can detect a fault (if any) within milliseconds and notify the RIP module of the fault. BFD for RIP can speed up fault detection and route convergence, which improves network reliability.

In BFD for RIP, BFD session establishment is triggered by RIP. When establishing a neighbor relationship, RIP will send detection parameters of the neighbor to BFD. Then, a BFD session will be established based on these detection parameters. If a link fault occurs, the local RIP process will receive a neighbor unreachable message within milliseconds. Then, the local RIP device will delete routing entries in which the neighbor relationship is Down and use the backup path to transmit messages.

Either of the following methods can be used to configure BFD for RIP:

- **Enable BFD in a RIP process:** This method is feasible when BFD for RIP needs to be enabled on most RIP interfaces.
- **Enable BFD on RIP interface:** This method is recommended when BFD for RIP needs to be enabled only on a small number of RIP interfaces.

 **NOTE**

The interface can be a physical interface or a GRE tunnel interface. If BFD is enabled on a GRE tunnel interface, millisecond-level fault detection can be implemented for the GRE tunnel.

## Pre-configuration Tasks

Before configuring BFD for RIP, complete the following tasks:

- Assign an IP address to each interface to ensure that neighboring nodes are reachable at the network layer.
- **Configuring Basic RIP Functions**

## Procedure

- Enable BFD in a RIP process.

a. Run **system-view**

The system view is displayed.

b. Run **bfd**

BFD is enabled globally.

c. Run **quit**

Return to the system view.

d. Run **rip process-id**

The RIP view is displayed.

e. Run **bfd all-interfaces enable**

BFD is enabled in the RIP process to establish BFD sessions.

If BFD is enabled globally, RIP will use default BFD parameters to establish BFD sessions on all the interfaces where RIP neighbor relationships are up in the process.

f. (Optional) Run **bfd all-interfaces { min-rx-interval min-receive-value | min-tx-interval min-transmit-value | detect-multiplier detect-multiplier-value } \***

The values of BFD parameters used to establish the BFD session are set.

Configure BFD parameter values based on the reliability requirements of the live network.

- If the live network requires high reliability, reduce the interval at which BFD packets are sent.
- If the live network requires low reliability, increase the interval at which BFD packets are sent.

BFD detection intervals are calculated as follows:

- Effective local interval at which BFD packets are sent = MAX { Configured local interval at which BFD packets are sent, Configured remote interval at which BFD packets are received }
- Effective local interval at which BFD packets are received = MAX { Configured remote interval at which BFD packets are sent, Configured local interval at which BFD packets are received }
- Effective local detection interval = Effective local interval at which BFD packets are received x Configured remote detection multiplier

Running the **bfd all-interfaces** command changes BFD session parameters on all RIP interfaces. The default detection multiplier and interval at which BFD packets are sent are recommended.

- g. (Optional) Perform the following operations to prevent an interface in the RIP process from establishing a BFD session:

- Run the **quit** command to return to the system view.
- Run the **interface interface-type interface-number** command to enter the view of a specified interface.
- Run the **rip bfd block** command to prevent the interface from establishing a BFD session.

- h. Run **commit**

The configuration is committed.

- Enable BFD on a RIP interface.

- a. Run **system-view**

The system view is displayed.

- b. Run **bfd**

BFD is enabled globally.

- c. Run **quit**

Return to the system view.

- d. Run **interface interface-type interface-number**

The view of the specified interface is displayed.

- e. Run **rip bfd enable**

BFD is enabled on the interface to establish a BFD session.

- f. (Optional) Run **rip bfd { min-rx-interval min-receive-value | min-tx-interval min-transmit-value | detect-multiplier detect-multiplier-value }**  
\*

The values of BFD parameters used to establish the BFD session are set.

- g. Run **commit**

The configuration is committed.

----End

## Checking the Configurations

After enabling BFD for RIP at both ends of a link, run the **display rip bfd session { interface interface-type interface-number | neighbor-id | all }** command. The following command output shows that the **BFDState** field on the local router is Up.

### 1.1.6.2.9 Configuring Static BFD for RIP

BFD provides link failure detection featuring light load and high speed. Static BFD for RIP is a mode to implement the BFD function.

## Context

Establishing BFD sessions between RIP neighbors enables the rapid detection of link faults and speeds up RIP's response to network topology changes. On a link that requires fast fault response and supports BFD on both ends, you can configure static BFD on both ends to implement common BFD detection.

To configure a static BFD session, you need to manually configure BFD detection using commands and deliver a BFD session establishment request.

## Pre-configuration Tasks

Before configuring static BFD for RIP, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring devices are reachable at the network layer.
- Configure basic RIP functions.**

## Data Preparation

To complete the configuration, you need the following data:

| No. | Data                                                    |
|-----|---------------------------------------------------------|
| 1   | ID of a RIP process                                     |
| 2   | Type and number of the interface to be enabled with BFD |

## Procedure

### Step 1 Enable BFD globally.

- Run the **system-view** command to enter the system view.
- Run the **bfd** command to enable BFD globally.
- Run the **quit** command to return to the system view.

### Step 2 Configure static BFD.

- Run the **bfd session-name bind peer-ip peer-ip [ interface interface-type interface-number ] [ source-ip source-ip ]** command to create BFD binding.

If both **peer-ip** (IP address of the peer end) and **interface** (local interface) are specified, BFD only monitors a single-hop link with **interface** as the outbound interface and **peer-ip** as the next hop address.

2. Set discriminators.

- To set the local discriminator, run the **discriminator local *discr-value*** command.
- To set the remote discriminator, run the **discriminator remote *discr-value*** command.

The local device's local discriminator and remote discriminator must be the same as the remote device's remote discriminator and local discriminator, respectively. Otherwise, a BFD session cannot be established.

 NOTE

Specifically, **local *discr-value*** of the local device must be the same as the **remote *discr-value*** of the remote device, and **remote *discr-value*** of the local device must be the same as **local *discr-value*** of the remote device.

3. Run the **quit** command to return to the system view.

**Step 3** Enable static BFD on an interface.

1. Run the **interface *interface-type interface-number*** command to enter the view of a specified interface.
2. Run the **rip bfd static** command to enable static BFD on the interface.
3. (Optional) Run the **rip bfd static binding *peer-address*** command to enable static BFD for the specified neighbor.

 NOTE

If you run both this command and the **rip bfd static** command, the latest configuration overrides the previous one. If the command is run, the existing BFD configuration mode will be changed.

**Step 4** Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

After configuring static BFD for RIP, run the **display rip process-id interface [*interface-type interface-number*] verbose** command to check the BFD for RIP configuration on a specified interface.

### 1.1.6.2.10 Configuring OPS to Control Packet Sending and Receiving on RIP Interfaces

This section describes how to prevent the failure of switching traffic to the backup path in a scenario where static unaffiliated BFD for RIP is configured and a link is available unidirectionally.

## Context

**Figure 1-208** shows an unaffiliated static BFD for RIP scenario where primary and backup paths exist. If the link from DeviceA to DeviceB on the primary path fails,

BFD on DeviceA rapidly detects the link fault, and the RIP route on DeviceA rapidly converges to the backup path. Because DeviceB does not support BFD or is not configured with BFD, DeviceB cannot rapidly detect the link fault. As a result, the RIP route on DeviceB cannot converge rapidly. In this case, DeviceB continues to send RIP packets to DeviceA. After DeviceA re-learns the RIP route from DeviceB, traffic is switched from the backup path to the primary path. However, the link from DeviceA to DeviceB is still faulty. As a result, traffic cannot be forwarded.

**Figure 1-208** Networking diagram of unaffiliated static BFD for RIP



Static BFD for RIP is enabled on DeviceA.

## Pre-configuration Tasks

Before configuring OPS to control packet sending and receiving on RIP interfaces, complete the following task:

- [Configure static BFD for RIP](#).

## Precautions

- The OPS mechanism allows a single script assistant to cache a maximum of 256 alarms. Therefore, this function does not apply to the scenario where BFD goes down due to a large number of link faults.
- After this function is enabled, you are advised not to perform an active/standby switchover, restart the device, or modify static BFD for RIP configurations when a link is faulty. Otherwise, the command for disabling an interface from receiving and sending RIP packets may fail to be deleted, causing a residual configuration.
- Broadcast network scenarios are not supported.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ops**

The OPS view is displayed.

### Step 3 Run **script-assistant python ripbfd\_down\_mtp.py**

A script assistant is created.

 NOTE

After this function is enabled, if BFD Down alarm information is received, the script assistant obtains information about the interface corresponding to the faulty link from the alarm information and determines whether to disable the interface from sending and receiving RIP packets based on whether the **rip bfd static** command is configured on the interface.

- If the **rip bfd static**, **undo rip output**, and **undo rip input** commands are run on the interface, the interface is disabled from sending and receiving RIP packets.
- If the **rip bfd static** command is not run on the interface, the interface is not affected.

**Step 4** Run **script-assistant python ripbfd\_up\_mtp.py**

A script assistant is created.

 NOTE

After this function is enabled, if BFD Down alarm clearing information is received, the script assistant obtains information about the interface corresponding to the faulty link from the alarm information and determines whether to allow the interface to send and receive RIP packets based on whether the **rip bfd static** command is configured on the interface.

- If the **rip bfd static**, **rip output**, and **rip input** commands are run on the interface, the interface is allowed to send and receive RIP packets.
- If the **rip bfd static** command is not run on the interface, the interface is not affected.

**Step 5** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the **display ops assistant current** command to check the current status of script assistants.

### 1.1.6.2.11 Improving the RIP Network Security

On a network requiring high security, you can configure RIP authentication and Generalized TTL Security Mechanism (GTSM) to improve network security.

## Usage Scenario

With the increase in attacks on TCP/IP networks and the defects in the TCP/IP protocol suite, network attacks have increasing impacts on the network security. Attacks on network devices may even cause a network crash or lead to network unavailability. Therefore, it is necessary to protect the network from attacks. RIP uses the following methods to ensure network security:

- RIP authentication: RIP checks the authentication mode and password in each packet to protect the local device against potential attacks.
- Check on the source IP address of each packet: RIP interfaces accept packets only from the same network to protect the local device from potential attacks from other networks.
- RIP GTSM: GTSM protects the local router by checking whether the time to live (TTL) value in the IP packet header is in a pre-defined range.

## Pre-configuration Tasks

Before improving RIP network security, complete the following tasks:

- Configure network layer addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.6.2.3 Configuring Basic RIP Functions](#)

## Configuration Procedure

Perform one or more of the following configuration tasks (excluding verifying the configuration) as required.

### Configuring an Authentication Mode for RIP-2 Packets

RIP-2 supports authentication for protocol packets. By default, authentication is not configured for RIP. Configuring authentication is recommended to ensure system security.

## Procedure

### Step 1 Run `system-view`

The system view is displayed.

### Step 2 Run `interface interface-type interface-number`

The interface view is displayed.

### Step 3 Perform the following operations as required:

- Run `rip authentication-mode simple { plain plain-text | [ cipher ] password-key }`  
Simple authentication is configured for RIP-2 packets.  
In simple authentication mode, the password in simple text mode is transmitted along with each authentication packet. Therefore, simple authentication is not recommended on networks requiring high security.
- Run `rip authentication-mode md5 { nonstandard { { plain plain-text | [ cipher ] password-key } key-id | keychain keychain-name } | usual { plain plain-text | [ cipher ] password-key } }`  
Message Digest 5 (MD5) authentication is configured for RIP-2 packets.  
In MD5 authentication mode, the MD5 password is used for packet encapsulation and decapsulation. MD5 authentication is more secure than simple authentication.

**nonstandard** supports nonstandard authentication packets.

**usual** supports Internet Engineering Task Force (IETF) standard authentication packets.

#### NOTE

For the sake of security, using the HMAC-SHA256 algorithm rather than the MD5 algorithm is recommended.

- Run `rip authentication-mode hmac-sha256 { plain plain-text | [ cipher ] password-key } key-id`

Hash Message Authentication Code for Secure Hash Algorithm 256 (HMAC-SHA256) authentication is configured.

### NOTICE

When configuring an authentication password, select the ciphertext mode because the password is saved in the configuration file as a plaintext if you select the plaintext mode, which has a high risk. To ensure device security, change the password periodically.

For security purposes, you are not advised to use the weak security algorithm in RIP. If you need to use the weak security algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

#### Step 4 Run commit

The configuration is committed.

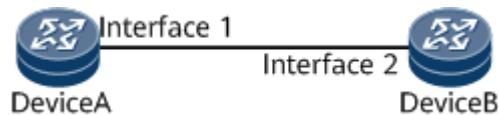
----End

### Result

The following section describes the changes in the RIP neighbor relationship and data traffic during the upgrade from RIP-1 to RIP-2 with authentication.

**Scenario 1:** [Figure 1-209](#) shows a P2P topology, and an upgrade is performed. DeviceA and DeviceB use the default configuration.

**Figure 1-209** RIP neighbor relationship over a P2P link



With the default configuration, packet sending and receiving on one end of the P2P link are the same as those on the other end.

Interface 1 on DeviceA: (Default version or RIP-1-compatible version)

Send: RIP-1

Receive: RIP-1; RIP-2 in broadcast or multicast mode

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)

Send: RIP-1

Receive: RIP-1; RIP-2 in broadcast or multicast mode

- Step 1: Configure authentication on interface 1 of DeviceA.

Interface 1 on DeviceA: (Default version or RIP-1-compatible version)

Send: RIP-1

Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information

DeviceB: (Default version or RIP-1-compatible version)

Send: RIP-1

Receive: RIP-1; RIP-2 in broadcast or multicast mode

Authentication applies to RIP-2 packets, not to RIP-1 packets; however, the packets exchanged between DeviceA and DeviceB are RIP-1 packets.

Therefore, the RIP neighbor relationship and data traffic remain unchanged after this step is performed.

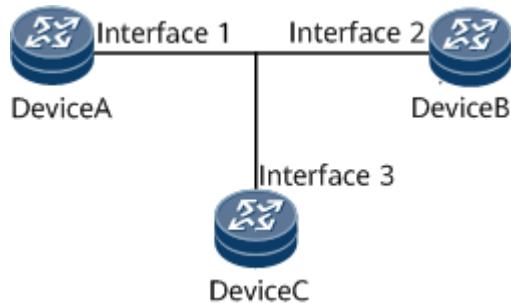
- Step 2: Configure the same authentication mode on interface 2 of DeviceB.  
Interface 1 on DeviceA: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
Authentication applies to RIP-2 packets, not to RIP-1 packets; however, the packets exchanged between DeviceA and DeviceB are RIP-1 packets. Therefore, the RIP neighbor relationship and data traffic remain unchanged after this step is performed.
- Step 3: Configure RIP-2 in broadcast mode on interface 1 of DeviceA.  
Interface 1 on DeviceA: (RIP-2 in broadcast mode)  
Send: RIP-2 in broadcast mode, with authentication information  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
DeviceA broadcasts RIP-2 packets with authentication information to DeviceB. Because the same authentication mode is configured on DeviceB, DeviceB accepts these packets. DeviceB sends RIP-1 packets without authentication information to DeviceA, which accepts the RIP-1 packets although it runs RIP-2 (in broadcast mode with authentication information).  
In this case, the neighbor relationship and data traffic are not affected.
- Step 4: Configure RIP-2 in broadcast mode on interface 2 of DeviceB.  
Interface 1 on DeviceA: (RIP-2 in broadcast mode)  
Send: RIP-2 in broadcast mode, with authentication information  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
Interface 2 on DeviceB: (RIP-2 in broadcast mode)  
Send: RIP-2 in broadcast mode, with authentication information  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
DeviceA and DeviceB exchange RIP-2 packets carrying authentication information in broadcast mode. Because authentication is configured on both DeviceA and DeviceB, both DeviceA and DeviceB can accept broadcast RIP-2 packets. In this case, the neighbor relationship and data traffic are not affected.
- Step 5: Configure RIP-2 on interface 1 of DeviceA.  
Interface 1 on DeviceA: (RIP-2 in broadcast mode)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in broadcast and multicast mode, with authentication information  
  
Interface 2 on DeviceB:  
Send: RIP-2 in broadcast mode, with authentication information  
Receive: RIP-1; RIP-2 in broadcast or multicast mode, with authentication information  
  
DeviceA and DeviceB exchange RIP-2 packets carrying authentication information in multicast mode. Because authentication has been configured on DeviceB, Device B accepts the packets from DeviceA. DeviceB broadcasts RIP-2 packets with authentication information to DeviceA, which accepts the packets although it runs RIP-2. In this case, the neighbor relationship and data traffic are not affected.
- Step 6: Configure RIP-2 on interface 2 of DeviceB.  
Interface 1 on DeviceA: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

Interface 2 on DeviceB: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

DeviceA and DeviceB multicast RIP-2 packets with authentication information to each other. Because authentication is configured on DeviceA and DeviceB, they can exchange the multicast RIP-2 packets. In this case, the neighbor relationship and data traffic are not affected.

**Scenario 2:** [Figure 1-210](#) shows a broadcast topology, and an upgrade from RIP-1 (or RIP-1-compatible version) to RIP-2 is performed on devices.

**Figure 1-210** RIP neighbor relationship over a broadcast link



Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode

Interface 1 on DeviceA: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Interface 3 on DeviceC: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

#### NOTE

All devices in the networking use interface-level RIP configuration.

- Step 1: Configure authentication on DeviceA.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode

Interface 1 on DeviceA: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Interface 3 on DeviceC: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Authentication applies to RIP-2 packets, not to RIP-1 packets. Therefore, the authentication on a RIP-1 interface does not affect RIP-1 packets.

- Step 2: Configure authentication on DeviceC.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode

Interface 1 on DeviceA: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Interface 3 on DeviceC: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Authentication applies to RIP-2 packets, not to RIP-1 packets. Therefore, the authentication on a RIP-1 interface does not affect RIP-1 packets.

- Step 3: Configure authentication on DeviceB.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Interface 3 on DeviceC: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

Authentication applies to RIP-2 packets, not to RIP-1 packets. Therefore, the authentication on an interface that runs RIP of the default version (RIP-1) does not affect RIP-1 packets. If RIP-2 packets are received, they are accepted only after they are authenticated. No RIP-2 packets are sent in this scenario. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 4: Run the **undo version** command on DeviceA.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (RIP-1)  
Send: RIP-1  
Receive: RIP-1

If the **undo rip version 1** command is run on the interface of DeviceB, the interface uses RIP of the default version (RIP-1) or RIP-1-compatible version. All the devices exchange RIP-1 packets in this scenario. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 5: Run the **undo version** command on DeviceC.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (Default version or RIP-1-compatible version)  
Send: RIP-1  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

If the **undo rip version 1** command is run on the interface of DeviceC, the interface uses RIP-1 (default version) or RIP-1-compatible version. All the devices exchange RIP-1 packets in this scenario. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 6: Configure RIP-2 in broadcast mode on DeviceA.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)  
Send: RIP-1

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (Default version or RIP-1-compatible version)

Send: RIP-1

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

If the **rip version 2 broadcast** command is run on the interface of DeviceA, the interface broadcasts RIP-2 packets with authentication information.

Because authentication is configured on other devices, they accept the RIP-2 packets from DeviceA and respond with RIP-1 packets. DeviceA accepts these RIP-1 packets. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 7: Configure RIP-2 in broadcast mode on DeviceC.

Interface 2 on DeviceB: (Default version or RIP-1-compatible version)

Send: RIP-1

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

If the **rip version 2 broadcast** command is run on the interface of DeviceC, the interface broadcasts RIP-2 packets with authentication information.

Because authentication is configured on other devices, they accept the RIP-2 packets from DeviceC. Device B still broadcasts RIP-1 packets. Upon receipt of these packets, DeviceA and DeviceC accept them. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 8: Configure RIP-2 in broadcast mode on DeviceB.

Interface 2 on DeviceB: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

If the **rip version 2 broadcast** command is run on the interface of DeviceB, the interface broadcasts RIP-2 packets with authentication information.

Because authentication is configured on other devices, they accept the RIP-2 packets from DeviceB and also broadcast RIP-2 packets with authentication information. All the devices accept packets from each other. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 9: Configure RIP-2 on DeviceA.

Interface 2 on DeviceB: (RIP-2 in broadcast mode)

Send: RIP-2 in broadcast mode, with authentication information

Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-2)

Send: RIP-2 in multicast mode, with authentication information

Receive: RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (RIP-2 in broadcast mode)  
Send: RIP-2 in broadcast mode, with authentication information  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

If the **rip version 2** command is run on the interface of DeviceA, the interface multicasts RIP-2 packets with authentication information. Because authentication is configured on other devices, they accept the RIP-2 packets from DeviceA and broadcast RIP-2 packets with authentication information. All the interfaces accept packets from each other. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 10: Configure RIP-2 on DeviceC.

Interface 2 on DeviceB: (RIP-2 in broadcast mode)  
Send: RIP-2 in broadcast mode, with authentication information  
Receive: RIP-1; RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

If the **rip version 2** command is run on the interface of DeviceC, the interface multicasts RIP-2 packets with authentication information. Because authentication is configured on other devices, they accept the RIP-2 packets from DeviceC. DeviceB broadcasts RIP-2 packets with authentication information, and DeviceA multicasts RIP-2 packets with authentication information. All the devices accept packets from each other. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

- Step 11: Configure RIP-2 on DeviceB.

Interface 2 on DeviceB: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

Interface 1 on DeviceA: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

Interface 3 on DeviceC: (RIP-2)  
Send: RIP-2 in multicast mode, with authentication information  
Receive: RIP-2 in multicast or broadcast mode, with authentication information

If the **rip version 2** command is run on the interface of DeviceB, the interface multicasts RIP-2 packets with authentication information. Because authentication is configured on other devices, they accept the RIP-2 packets from DeviceB and multicast RIP-2 packets with authentication information. All the devices accept packets from each other. Therefore, the RIP neighbor relationships and data traffic remain unchanged after this step is performed.

### Scenario 3: RIP-2 or RIP-2 in broadcast mode is enabled on all devices.

In this scenario, if RIP authentication is configured on all devices, packets may be discarded in the following cases because authentication configuration synchronization cannot be ensured during authentication:

1. RIP packets with authentication information are received by interfaces without authentication configuration.
2. RIP packets without authentication information are received by interfaces with authentication configuration.

After all the configurations are performed, RIP packets are authenticated and accepted by all interfaces with authentication configuration. When the default timer is used, RIP considers a route invalid only if it does not receive any Update packet within six update intervals. Therefore, if all the configurations are completed within the six update intervals, no traffic is interrupted.

## Configuring the Check on the Source Address in RIP Packets on the Broadcast Network

By default, RIP checks the source addresses of received packets, and the local RIP interface receives only the packets from the same network.

### Context

RIP interfaces check the source address in received RIP packets. If the source address of the received RIP packet is from a network segment different from that of the local interface IP address, the local interface discards this RIP packet. This improves the network security.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

#### Step 3 Run **verify-source**

The source address check is configured for RIP packets on the broadcast network.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring GTSM for RIP

Generalized TTL Security Mechanism (GTSM) defends against attacks by checking the TTL value.

### Context

During network attacks, attackers may simulate RIP packets and continuously send them to a router. If the packets are destined for the router, it directly forwards them to the control plane for processing without validating them. As a result, the increased processing workload on the control plane results in high CPU usage. Generalized TTL Security Mechanism (GTSM) defends against attacks by checking whether the time to live (TTL) value in each IP packet header is within a pre-defined range.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **rip valid-ttl-hops valid-ttl-hops-value [ vpn-instance vpn-instance-name ]**

GTSM is configured for RIP.



The valid TTL range of the detected packets is [ 255 -*valid-ttl-hops-value* + 1, 255 ].

### Step 3 Run **commit**

The configuration is committed.

### Step 4 Set the default action for packets that do not match the GTSM policy.

GTSM only checks the TTL values of packets that match the GTSM policy. Packets that do not match the GTSM policy can be allowed or dropped.

You can enable the log function to record packet drop for troubleshooting.

Perform the following configurations on the GTSM-enabled router:

#### 1. Run **system-view**

The system view is displayed.

#### 2. Run **gtsm default-action { drop | pass }**

The default action for packets that do not match the GTSM policy is configured.



If the default action is configured but no GTSM policy is configured, GTSM does not take effect.

This command is supported only on the Admin-VS and cannot be configured in other VSs. This command takes effect on all VSs.

#### 3. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the security function of RIP is successfully configured, you can view information about RIP interfaces and the current running status of RIP.

## Prerequisites

The configurations for enhancing RIP network security have been performed.

## Procedure

- Run the **display rip process-id interface [ interface-type interface-number ] [ verbose ]** command to check information about RIP interfaces.

- Run the **display rip process-id** command to check the running status and configurations of the specified RIP process.

----End

### 1.1.6.2.12 Configuring RIP Network Management

By binding RIP and MIBs, you can view and configure RIP through the NMS.

#### Usage Scenario

Through Simple Network Management Protocol (SNMP), the RIP Management Information Base (MIB) manages multicast information exchanged between the NMS and agents.

#### Pre-configuration Tasks

Before controlling RIP configuration through an SNMP agent, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.6.2.3 Configuring Basic RIP Functions](#)

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **rip mib-binding process-id**

The MIB is bound to the RIP process ID, and the ID of the RIP process that accepts SNMP requests is specified.

##### Step 3 Run **commit**

The configuration is committed.

----End

#### Checking the Configurations

Run the following commands to check the previous configuration.

- Run the **display current-configuration** command to check the valid parameters on the device.

### 1.1.6.2.13 Maintaining RIP

RIP maintenance is implemented through debugging which affects system performance.

#### Clearing RIP

If you want to clear information about RIP, run the reset commands in the user view.

## Context

### NOTICE

RIP information cannot be restored after you clear it. Therefore, exercise caution when running the following commands.

## Procedure

- Run the **reset rip { process-id| all } configuration** command to reset the configuration parameters of the specific RIP process. When the RIP process is initiated, the default configuration parameters are restored to the default values.
- Run the **reset rip { process-id| all } imported-routes** command to clear the routes imported from other routing protocols and then import these routes into RIP again.
- Run the **reset rip { process-id| all } statistics [ interface interface-type interface-number ]** command to clear the statistics of the counter that is maintained by a particular RIP process. This command helps to re-collect statistics during debugging.

----End

## Debugging RIP

Routers can generate associated debugging information after you enable the debugging of modules in the user view. Debugging information displays the contents of the packets sent or received by the debugged modules.

## Context

### NOTICE

Debugging affects system performance. Therefore, after debugging, run the **undo debugging all** command to disable it immediately.

If a RIP fault occurs, run the **debugging** command in the user view to debug RIP, view the debugging information, and locate and analyze the fault.

## Procedure

- Step 1** Run the **debugging rip process-id packet [ send | receive ] [ error ] [ verbose ] [ interface interface-type interface-number [ peer peer-address ] ]** command to debug the RIP packets sent or received on the network.
- Step 2** Run the **debugging rip process-id route [ error | backup ] [ imported ] { interface interface-type interface-number [ peer peer-address ] } ]** command to debug RIP routes.

**Step 3** Run the **debugging rip miscellaneous** command to debug RIP packets globally.

----End

#### 1.1.6.2.14 Configuration Examples for RIP

This section provides RIP configuration examples.

#### Example for Configuring Basic RIP Functions

This section describes how to configure basic RIP functions, including how to enable RIP and configure a RIP version number on each device.

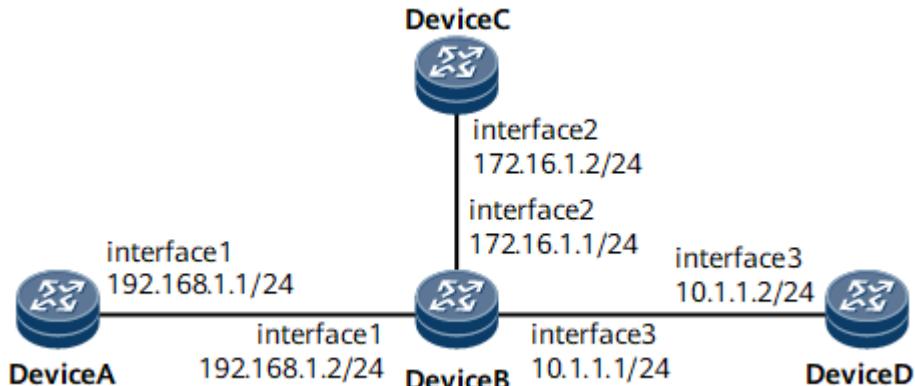
#### Networking Requirements

On the network shown in [Figure 1-211](#), it is required that RIP be enabled on all interfaces of DeviceA, DeviceB, DeviceC, and DeviceD and that these interfaces communicate with each other through RIP-2.

**Figure 1-211** Configuring basic RIP functions



In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



#### Precautions

During the configuration, note the following:

- Different network segments on the same physical interface cannot be enabled in different RIP processes.
- If a RIP process is bound to a VPN instance, interfaces in this RIP process also need to be bound to the VPN instance.

#### Configuration Roadmap

The configuration roadmap is as follows:

1. Assign an IP address to each interface to ensure network connectivity.

2. Configure RIP-2 packet authentication.
3. Enable RIP and configure basic RIP functions on each router.
4. Configure RIP-2 on each router and check the subnet mask.

## Data Preparation

To complete the configuration, you need the following data:

- RIP-enabled network segment 192.168.1.0 on DeviceA
- RIP-enabled network segments 192.168.1.0, 172.16.0.0, and 10.0.0.0 on DeviceB
- RIP-enabled network segment 172.16.0.0 on DeviceC
- RIP-enabled network segment 10.0.0.0 on DeviceD
- RIP-2 on DeviceA, DeviceB, DeviceC, and DeviceD

## Procedure

**Step 1** Assign an IP address to each interface. For configuration details, see configuration files in this section.

**Step 2** Configure RIP-2 packet authentication.

# Configure DeviceA.

```
[~DeviceA] interface gigabitethernet1/0/0
[*DeviceA-GigabitEthernet1/0/0] rip authentication-mode hmac-sha256 cipher YsHsjx_202206 200
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA] quit
```

The configurations of other devices are similar to the configuration of DeviceA. For configuration details, see configuration files.

### NOTE

The directly connected interfaces must be configured with the same authentication password. Otherwise, the neighbor relationship cannot be established.

**Step 3** Configure the network segments to be enabled with RIP.

# Configure DeviceA.

```
[~DeviceA] rip
[*DeviceA-rip-1] network 192.168.1.0
[*DeviceA-rip-1] commit
[~DeviceA-rip-1] quit
```

# Configure DeviceB.

```
[~DeviceB] rip
[*DeviceB-rip-1] network 192.168.1.0
[*DeviceB-rip-1] network 172.16.0.0
[*DeviceB-rip-1] network 10.0.0.0
[*DeviceB-rip-1] commit
[~DeviceB-rip-1] quit
```

# Configure DeviceC.

```
[~DeviceC] rip
[*DeviceC-rip-1] network 172.16.0.0
[*DeviceC-rip-1] commit
```

```
[~DeviceC-rip-1] quit

Configure DeviceD.

[~DeviceD] rip
[*DeviceD-rip-1] network 10.0.0.0
[*DeviceD-rip-1] commit
[~DeviceD-rip-1] quit
```

# Check the RIP routing table of DeviceA.

```
[~DeviceA] display rip 1 route
Route Flags: R - RIP
 A - Aging, S - Suppressed, G - Garbage-collect

Peer 192.168.1.2 on GigabitEthernet1/0/0
 Destination/Mask Nexthop Cost Tag Flags Sec
 10.0.0.0/8 192.168.1.2 1 0 RA 14
 172.16.0.0/16 192.168.1.2 1 0 RA 14
```

The command output shows that the routes advertised by RIP-1 use natural masks.

**Step 4** Configure the RIP version number.

# Configure RIP-2 on DeviceA.

```
[~DeviceA] rip
[~DeviceA-rip-1] version 2
[*DeviceA-rip-1] commit
[~DeviceA-rip-1] quit
```

# Configure RIP-2 on DeviceB.

```
[~DeviceB] rip
[~DeviceB-rip-1] version 2
[*DeviceB-rip-1] commit
[~DeviceB-rip-1] quit
```

# Configure RIP-2 on DeviceC.

```
[~DeviceC] rip
[~DeviceC-rip-1] version 2
[*DeviceC-rip-1] commit
[~DeviceC-rip-1] quit
```

# Configure RIP-2 on DeviceD.

```
[~DeviceD] rip
[~DeviceD-rip-1] version 2
[*DeviceD-rip-1] commit
[~DeviceD-rip-1] quit
```

**Step 5** Verify the configuration.

# Check the RIP routing table of DeviceA.

```
[~DeviceA] display rip 1 route
Route Flags: R - RIP
 A - Aging, S - Suppressed, G - Garbage-collect

Peer 192.168.1.2 on GigabitEthernet1/0/0
 Destination/Mask Nexthop Cost Tag Flags Sec
 10.1.1.0/24 192.168.1.2 1 0 RA 32
 172.16.1.0/24 192.168.1.2 1 0 RA 32
```

The command output shows that the routes advertised by RIP-2 contain more accurate subnet masks.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
rip authentication-mode hmac-sha256 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^%# 200

interface LoopBack1
ip address 1.1.1.1 255.255.255.255
rip enable 1

rip 1
version 2
network 192.168.1.0

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
rip authentication-mode hmac-sha256 cipher %^%#*&/]"$OoC.u#h5%iA0Q.3,$mP[]0;lvk-,Gyy/w4%^%# 200

interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
rip authentication-mode hmac-sha256 cipher %^%#BVZr*tW;4"],!H~L*XPyb.Y!BVdHE`D,uM~1q"<%^%# 200

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
rip authentication-mode hmac-sha256 cipher %^%#D[hPG]hUbHNR-EJM.
%>P&OR}NU]W&L>GAd84)-7,%^%# 200

interface LoopBack1
ip address 2.2.2.2 255.255.255.255
rip enable 1

rip 1
version 2
network 10.0.0.0
network 172.16.0.0
network 192.168.1.0

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet2/0/0
undo shutdown
```

```
ip address 172.16.1.2 255.255.255.0
rip authentication-mode hmac-sha256 cipher %^
%#ogrEOF0J;)8umQDUCfm8uc92G2xV@By=^#;<~2zF%^%# 200
#
interface LoopBack1
ip address 3.3.3.3 255.255.255.255
rip enable 1
#
rip 1
version 2
network 172.16.0.0
#
return
```

- DeviceD configuration file

```

sysname DeviceD

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
rip authentication-mode hmac-sha256 cipher %^%#+40P:fm"RB,[>]'<6O1B"!K[Go_=u4Q]Yp$Hh:wJ%^%# 200
#
interface LoopBack1
ip address 4.4.4.4 255.255.255.255
rip enable 1
#
rip 1
version 2
network 10.0.0.0
#
return
```

## Example for Preventing Routing Loops

This section describes how to configure split horizon to prevent routing loops.

### Networking Requirements

On the network shown in [Figure 1-212](#), IP addresses have been configured for interfaces on all routers, RIP-2 has been configured on each router, and RIP services are running properly. Classful summarization is enabled on routerA and routerC. It is required that split horizon be configured on routerA and routerC.

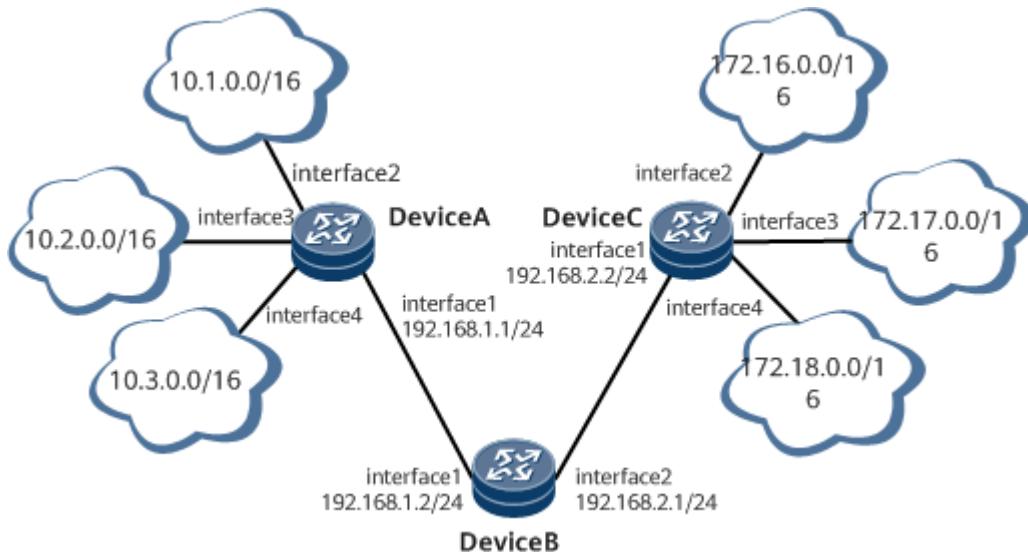
#### NOTE

When configuring classful summarization on a RIP-2 network, you need to disable split horizon and poison reverse. After classful summarization is disabled, you need to reconfigure split horizon or poison reverse to prevent routing loops. This section uses split horizon as an example.

**Figure 1-212** Network diagram of configuring routing loop prevention

#### NOTE

interface1, interface2, interface3, and interface4 in this example represent GE1/0/0, GE1/0/1, GE1/0/2, and GE1/0/3, respectively.



## Precautions

During the configuration, note the following:

If both split horizon and poison reverse are configured, only poison reverse takes effect.

To improve security, you are advised to configure RIP-2 packet authentication. For details, see "Improving RIP Network Security." The following example describes how to configure an authentication mode for RIP-2 packets. For details, see "Example for Configuring Basic RIP Functions."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Disable route summarization.
2. Enable split horizon.

## Data Preparation

To complete the configuration, you need the following data:

- RIP-enabled network segments 10.1.0.0, 10.2.0.0, 10.3.0.0, and 192.168.0.0 on DeviceA
- RIP-enabled network segment 192.168.0.0 on DeviceB
- RIP-enabled network segments 172.16.0.0, 172.17.0.0, 172.18.0.0, and 192.168.0.0 on DeviceC
- IP addresses of interfaces

## Procedure

### Step 1 Disable route summarization.

# Disable route summarization on DeviceA.

```
[~DeviceA] rip 1
```

```
[*DeviceA-rip-1] undo summary
[*DeviceA-rip-1] commit
[~DeviceA-rip-1] quit
```

# Disable route summarization on DeviceB.

```
[~DeviceB] rip 1
[*DeviceB-rip-1] undo summary
[*DeviceB-rip-1] commit
[~DeviceB-rip-1] quit
```

## Step 2 Configure split horizon.

Configure split horizon on the RIP interfaces of all routers. The configurations of DeviceB and DeviceC are similar to the configuration of DeviceA. For details, see configuration files.

# Configure DeviceA.

```
[~DeviceA] interface gigabitethernet1/0/0
[~DeviceA-GigabitEthernet1/0/0] rip split-horizon
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet1/0/1
[*DeviceA-GigabitEthernet1/0/1] rip split-horizon
[*DeviceA-GigabitEthernet1/0/1] quit
[*DeviceA] interface gigabitethernet1/0/2
[*DeviceA-GigabitEthernet1/0/2] rip split-horizon
[*DeviceA-GigabitEthernet1/0/2] quit
[*DeviceA] interface gigabitethernet1/0/3
[*DeviceA-GigabitEthernet1/0/3] rip split-horizon
[*DeviceA-GigabitEthernet1/0/3] quit
[*DeviceA] commit
```

## Step 3 Verify the configuration.

# Run the **display rip 1 interface verbose** command on DeviceA and DeviceC to check the result after split horizon is configured. Use the command output on DeviceA as an example. If the value of the Split-Horizon field is Enabled, split horizon has been enabled.

```
[~DeviceA] display rip 1 interface verbose
GigabitEthernet1/0/0(192.168.1.1)
State : DOWN MTU: 500
Metricin : 0
Metricout: 1
Input : Enabled Output : Enabled
Protocol : RIPv2 Multicast
Send : RIPv2 Multicast Packets
Receive : RIPv2 Multicast and Broadcast Packets
Poison-reverse : Disabled
Split-Horizon : Enabled
Authentication type : None
Replay Protection : Disabled
Max Packet Length : 512
GigabitEthernet1/0/1(10.1.1.1)
State : DOWN MTU: 500
Metricin : 0
Metricout: 1
Input : Enabled Output : Enabled
Protocol : RIPv2 Multicast
Send : RIPv2 Multicast Packets
Receive : RIPv2 Multicast and Broadcast Packets
Poison-reverse : Disabled
Split-Horizon : Enabled
Authentication type : None
Replay Protection : Disabled
Max Packet Length : 512
GigabitEthernet1/0/2(10.2.1.1)
```

```
State : DOWN MTU: 500
Metricin : 0
Metricout: 1
Input : Enabled Output : Enabled
Protocol : RIPv2 Multicast
Send : RIPv2 Multicast Packets
Receive : RIPv2 Multicast and Broadcast Packets
Poison-reverse : Disabled
Split-Horizon : Enabled
Authentication type : None
Replay Protection : Disabled
Max Packet Length : 512
GigabitEthernet1/0/3(10.3.1.1)
State : DOWN MTU: 500
Metricin : 0
Metricout: 1
Input : Enabled Output : Enabled
Protocol : RIPv2 Multicast
Send : RIPv2 Multicast Packets
Receive : RIPv2 Multicast and Broadcast Packets
Poison-reverse : Disabled
Split-Horizon : Enabled
Authentication type : None
Replay Protection : Disabled
Max Packet Length : 512
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
rip version 2 multicast

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.1.1 255.255.0.0
rip version 2 multicast

interface GigabitEthernet1/0/2
undo shutdown
ip address 10.2.1.1 255.255.0.0
rip version 2 multicast

interface GigabitEthernet1/0/3
undo shutdown
ip address 10.3.1.1 255.255.0.0
rip version 2 multicast

rip 1
network 10.1.0.0
network 10.2.0.0
network 10.3.0.0
network 192.168.0.0
undo summary
return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
```

```
ip address 192.168.1.2 255.255.255.0
rip version 2 multicast
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 192.168.2.1 255.255.255.0
rip version 2 multicast
#
rip 1
network 192.168.0.0
undo summary
return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0
rip version 2 multicast

interface GigabitEthernet1/0/1
undo shutdown
ip address 172.16.1.1 255.255.0.0
rip version 2 multicast

interface GigabitEthernet1/0/2
undo shutdown
ip address 172.17.1.1 255.255.0.0
rip version 2 multicast

interface GigabitEthernet1/0/3
undo shutdown
ip address 172.18.1.1 255.255.0.0
rip version 2 multicast
#
rip 1
network 172.16.0.0
network 172.17.0.0
network 172.18.0.0
network 192.168.0.0
return
```

## Example for Configuring RIP to Import Routes

This section describes how to configure RIP to import external routes to increase the number of routes in the RIP routing table.

## Networking Requirements

On the network shown in [Figure 1-213](#), DeviceB runs two RIP processes: RIP 100 and RIP 200. DeviceB exchanges routing information with DeviceA through RIP 100 and with DeviceC through RIP 200.

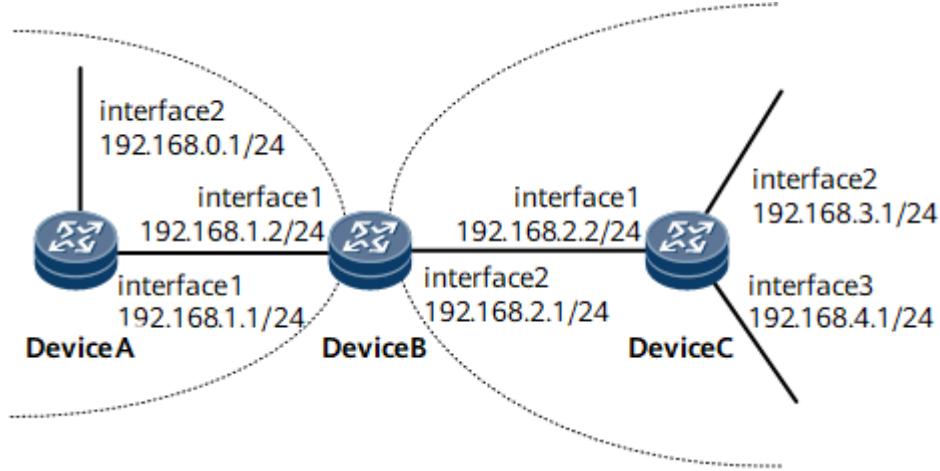
It is required that DeviceB be configured to import routes from two processes to each other and that the default metric of the routes imported from RIP 200 be set to 3.

In addition, a filtering policy needs to be configured on DeviceB to filter out the route 192.168.4.0/24 imported from RIP 200 so that the route is not advertised to DeviceA.

Figure 1-213 Network diagram of configuring RIP to import external routes

 NOTE

In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Context

Run one of the following commands to set a metric for imported routes. The commands are listed in descending order of priority:

- **apply cost**: sets a route metric.
- **import-route (RIP view)**: sets a metric for imported routes.
- **default-cost (RIP view)**: sets a metric for default routes.

## Precautions

To improve security, you are advised to configure RIP-2 packet authentication. For details, see "Improving RIP Network Security." The following example describes how to configure an authentication mode for RIP-2 packets. For details, see "Example for Configuring Basic RIP Functions."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable RIP 100 and RIP 200 on each router and specify network segments.
2. Configure two RIP processes on DeviceB to import routes from each other and set the default metric of the routes imported from RIP 200 to 3.
3. Configure an ACL on DeviceB to filter out the routes imported from RIP 200.

## Data Preparation

To complete the configuration, you need the following data:

- RIP network segments 192.168.0.0 and 192.168.1.0 on DeviceA
- RIP network segments 192.168.1.0 and 192.168.2.0 on DeviceB

- RIP network segments 192.168.2.0, 192.168.3.0, and 192.168.4.0 on DeviceC

## Procedure

**Step 1** Assign an IP address to each interface. For configuration details, see configuration files in this section.

**Step 2** Configure basic RIP functions.

# Enable RIP process 100 on DeviceA.

```
[~DeviceA] rip 100
[*DeviceA-rip-100] network 192.168.0.0
[*DeviceA-rip-100] network 192.168.1.0
[*DeviceA-rip-100] commit
[~DeviceA-rip-100] quit
```

# Enable RIP 100 and RIP 200 on DeviceB.

```
[~DeviceB] rip 100
[*DeviceB-rip-100] network 192.168.1.0
[*DeviceB-rip-100] quit
[*DeviceB] rip 200
[*DeviceB-rip-200] network 192.168.2.0
[*DeviceB-rip-200] commit
[~DeviceB-rip-200] quit
```

# Enable RIP process 200 on DeviceC.

```
[~DeviceC] rip 200
[*DeviceC-rip-200] network 192.168.2.0
[*DeviceC-rip-200] network 192.168.3.0
[*DeviceC-rip-200] network 192.168.4.0
[*DeviceC-rip-200] commit
[~DeviceC-rip-200] quit
```

# Check the routing table of DeviceA.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: _public_
Destinations : 7 Routes : 7
Destination/Mask Proto Pre Cost Flags NextHop Interface
 127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
 127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
 192.168.0.0/24 Direct 0 0 D 192.168.0.1 GigabitEthernet2/0/0
 192.168.0.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
 192.168.1.0/24 Direct 0 0 D 192.168.1.1 GigabitEthernet1/0/0
 192.168.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 192.168.1.2/32 Direct 0 0 D 192.168.1.2 GigabitEthernet1/0/0
```

The command output shows that the routing table of DeviceA does not contain the routes of other processes.

**Step 3** Configure RIP to import external routes.

# Set the default route metric to 3 on DeviceB and import routes of the two RIP processes into the routing table of each other.

```
[~DeviceB] rip 100
[*DeviceB-rip-100] default-cost 3
[*DeviceB-rip-100] import-route rip 200
[*DeviceB-rip-100] quit
[*DeviceB] rip 200
[*DeviceB-rip-200] import-route rip 100
[*DeviceB-rip-200] quit
```

[\*DeviceB] **commit**

# Check the routing table of DeviceA after the routes are imported.

[~DeviceA] **display ip routing-table**

Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: \_public\_

Destinations : 10 Routes : 10

| Destination/Mask      | Proto  | Pre | Cost | Flags | NextHop            | Interface                   |
|-----------------------|--------|-----|------|-------|--------------------|-----------------------------|
| 127.0.0.0/8           | Direct | 0   | 0    | D     | 127.0.0.1          | InLoopBack0                 |
| 127.0.0.1/32          | Direct | 0   | 0    | D     | 127.0.0.1          | InLoopBack0                 |
| 192.168.0.0/24        | Direct | 0   | 0    | D     | 192.168.0.1        | GigabitEthernet2/0/0        |
| 192.168.0.1/32        | Direct | 0   | 0    | D     | 127.0.0.1          | GigabitEthernet2/0/0        |
| 192.168.1.0/24        | Direct | 0   | 0    | D     | 192.168.1.1        | GigabitEthernet1/0/0        |
| 192.168.1.1/32        | Direct | 0   | 0    | D     | 127.0.0.1          | GigabitEthernet1/0/0        |
| 192.168.1.2/32        | Direct | 0   | 0    | D     | 192.168.1.2        | GigabitEthernet1/0/0        |
| <b>192.168.2.0/24</b> | RIP    | 100 | 4    | D     | <b>192.168.1.2</b> | <b>GigabitEthernet1/0/0</b> |
| 192.168.3.0/24        | RIP    | 100 | 4    | D     | 192.168.1.2        | GigabitEthernet1/0/0        |
| 192.168.4.0/24        | RIP    | 100 | 4    | D     | 192.168.1.2        | GigabitEthernet1/0/0        |

The RIP routing table of DeviceA contains routes 192.168.2.0/24, 192.168.3.0/24, and 192.168.3.0/24. These new routes are learned from RIP process 200 on DeviceB.

#### Step 4 Configure RIP to filter imported routes.

# Configure an ACL on DeviceB and set a rule to deny the packets with source address 192.168.4.0/24.

[~DeviceB] **acl 2000**

[\*DeviceB-acl4-basic-2000] **rule deny source 192.168.4.0 0.0.0.255**

[\*DeviceB-acl4-basic-2000] **rule permit**

[\*DeviceB-acl4-basic-2000] **quit**

# On DeviceB, filter out the route 192.168.4.0/24 imported from RIP 200 according to the ACL rule.

[\*DeviceB] **rip 100**

[\*DeviceB-rip-100] **filter-policy 2000 export**

[\*DeviceB-rip-100] **quit**

[\*DeviceB] **commit**

#### Step 5 Verify the configuration.

# Check the routing table of DeviceA after the filtering.

[~DeviceA] **display ip routing-table**

Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table: \_public\_

Destinations : 9 Routes : 9

| Destination/Mask | Proto  | Pre | Cost | Flags | NextHop     | Interface            |
|------------------|--------|-----|------|-------|-------------|----------------------|
| 127.0.0.0/8      | Direct | 0   | 0    | D     | 127.0.0.1   | InLoopBack0          |
| 127.0.0.1/32     | Direct | 0   | 0    | D     | 127.0.0.1   | InLoopBack0          |
| 192.168.0.0/24   | Direct | 0   | 0    | D     | 192.168.0.1 | GigabitEthernet2/0/0 |
| 192.168.0.1/32   | Direct | 0   | 0    | D     | 127.0.0.1   | GigabitEthernet2/0/0 |
| 192.168.1.0/24   | Direct | 0   | 0    | D     | 192.168.1.1 | GigabitEthernet1/0/0 |
| 192.168.1.1/32   | Direct | 0   | 0    | D     | 127.0.0.1   | GigabitEthernet1/0/0 |
| 192.168.1.2/32   | Direct | 0   | 0    | D     | 192.168.1.2 | GigabitEthernet1/0/0 |
| 192.168.2.0/24   | RIP    | 100 | 4    | D     | 192.168.1.2 | GigabitEthernet1/0/0 |
| 192.168.3.0/24   | RIP    | 100 | 4    | D     | 192.168.1.2 | GigabitEthernet1/0/0 |

The command output shows that the RIP routing table of DeviceA changes. That is, the route with the source address 192.168.4.0/24 is denied.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0

rip 100
network 192.168.0.0
network 192.168.1.0

return
```

- DeviceB configuration file

```

sysname DeviceB

acl number 2000
rule 5 deny source 192.168.4.0 0.0.0.255
rule 10 permit

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.1 255.255.255.0

rip 100
default-cost 3
network 192.168.1.0
filter-policy 2000 export
import-route rip 200

rip 200
network 192.168.2.0
import-route rip 100

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.3.1 255.255.255.0

interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.4.1 255.255.255.0

rip 200
network 192.168.2.0
network 192.168.3.0
network 192.168.4.0
#
```

return

## Example for Configuring Dynamic BFD for RIP

This section describes how to configure dynamic BFD on a RIP network to rapidly detect link faults and notify RIP of the faults to trigger service traffic switchover.

## Networking Requirements

RIP periodically exchanges Update messages to monitor the status of neighbors. By default, if a local device does not receive any Update messages from its neighbor after six update intervals (180s) elapse, it considers the neighbor down.

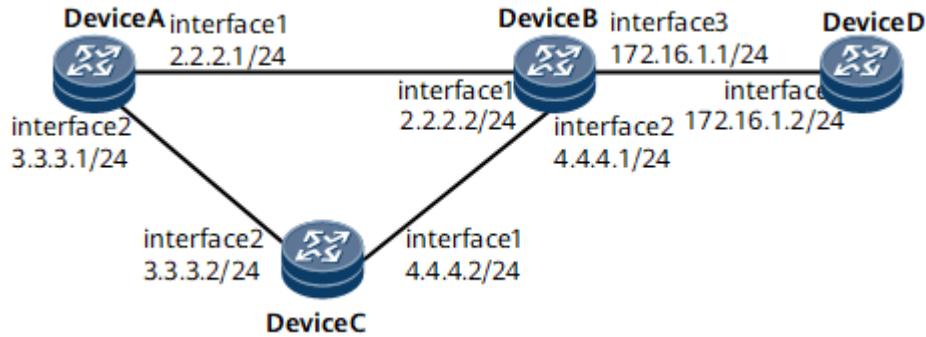
Voice, video, and other VOD services are widely used and are sensitive to the packet loss and delay. Long-time fault detection will cause the loss of a large number of data packets. As a result, the requirement of carrier-class networks for high reliability cannot be met. BFD for RIP can be deployed to complete link fault detection within milliseconds, speeding up RIP convergence.

On the network shown in [Figure 1-214](#), primary and backup links are deployed. The primary link is Device A -> Device B, and the backup link is Device A -> Device C -> Device B. In normal cases, service traffic is transmitted along the primary link. If the primary link fails, it is expected that the fault is rapidly detected and that service traffic is switched to the backup link in time. In this case, you can configure BFD for RIP to monitor the RIP neighbor relationship between DeviceA and DeviceB. If the link between DeviceA and DeviceB fails, BFD can rapidly detect the fault and notify RIP of the fault. Then service traffic is switched to the backup link for transmission.

**Figure 1-214** Network diagram of configuring dynamic BFD for RIP

### NOTE

In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

To improve security, you are advised to configure RIP-2 packet authentication. For details, see "Improving RIP Network Security." The following example describes how to configure an authentication mode for RIP-2 packets. For details, see "Example for Configuring Basic RIP Functions."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable basic RIP functions on each router and ensure that RIP neighbor relationships are established.
2. Enable global BFD.
3. Enable BFD on the interfaces of the link between DeviceA and DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

- DeviceA: RIP process ID (1), RIP version number (2), IP address of GE 1/0/0 (2.2.2.1/24), and IP address of GE 2/0/0 (3.3.3.1/24)
- DeviceB: RIP process ID (1), RIP version number (2), IP address of GE 1/0/0 (2.2.2.2/24), IP address of GE 2/0/0 (4.4.4.1/24), and IP address of GE 3/0/0 (172.16.1.1/24)
- DeviceC: RIP process ID (1), RIP version number (2), IP address of GE 1/0/0 (4.4.4.2/24), and IP address of GE 2/0/0 (3.3.3.2/24)
- DeviceD: RIP process ID (1), RIP version number (2), and IP address of GE 1/0/0 (172.16.1.2/24)
- Minimum interval at which BFD packets are sent (100 ms), minimum interval at which BFD packets are received (100 ms), and local detection multiplier (10) on DeviceA and DeviceB

## Procedure

### Step 1 Configure IP addresses for interfaces.

Assign an IP address to each interface according to [Figure 1-214](#) and [Data Preparation](#). For configuration details, see configuration files in this section.

### Step 2 Configure basic RIP functions.

# Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] rip 1
[*DeviceA-rip-1] version 2
[*DeviceA-rip-1] network 2.0.0.0
[*DeviceA-rip-1] network 3.0.0.0
[*DeviceA-rip-1] commit
[~DeviceA-rip-1] quit
```

# Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] rip 1
[*DeviceB-rip-1] version 2
[*DeviceB-rip-1] network 2.0.0.0
[*DeviceB-rip-1] network 4.0.0.0
[*DeviceB-rip-1] network 172.16.0.0
[*DeviceB-rip-1] commit
[~DeviceB-rip-1] quit
```

# Configure DeviceC.

```
<DeviceC> system-view
```

```
[~DeviceC] rip 1
[*DeviceC-rip-1] version 2
[*DeviceC-rip-1] network 3.0.0.0
[*DeviceC-rip-1] network 4.0.0.0
[*DeviceC-rip-1] commit
[~DeviceC-rip-1] quit
```

# Configure DeviceD.

```
<DeviceD> system-view
[~DeviceD] rip 1
[*DeviceD-rip-1] version 2
[*DeviceD-rip-1] network 172.16.0.0
[*DeviceD-rip-1] commit
[~DeviceD-rip-1] quit
```

# After completing the preceding configurations, run the **display rip neighbor** command. The command output shows that DeviceA has established neighbor relationships with DeviceB and DeviceC. Use the command output on DeviceA as an example.

```
[~DeviceA] display rip 1 neighbor
```

| IP Address              | Interface Type       | Type | Last-Heard-Time |
|-------------------------|----------------------|------|-----------------|
| 3.3.3.2                 | GigabitEthernet2/0/0 | RIP  | 0:0:5           |
| Number of RIP routes :2 |                      |      |                 |
| 2.2.2.2                 | GigabitEthernet1/0/0 | RIP  | 0:0:5           |
| Number of RIP routes :4 |                      |      |                 |

# Run the **display ip routing-table** command. The command output shows that routers can import routes from each other. Take the command output on DeviceA as an example.

```
[~DeviceA] display ip routing-table
```

```
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table : _public_
Destinations : 15 Routes : 15
```

| Destination/Mask   | Proto  | Pre | Cost | Flags | NextHop   | Interface            |
|--------------------|--------|-----|------|-------|-----------|----------------------|
| 3.0.0.0/8          | RIP    | 100 | 3    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
| 3.3.3.0/24         | Direct | 0   | 0    | D     | 3.3.3.1   | GigabitEthernet2/0/0 |
| 3.3.3.1/32         | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 3.3.3.255/32       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 2.0.0.0/8          | RIP    | 100 | 3    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 2.2.2.0/24         | Direct | 0   | 0    | D     | 2.2.2.1   | GigabitEthernet1/0/0 |
| 2.2.2.1/32         | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 2.2.2.255/32       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 127.0.0.0/8        | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32       | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.255.255.255/32 | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 172.16.0.0/16      | RIP    | 100 | 4    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 172.16.1.0/24      | RIP    | 100 | 1    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 4.4.4.0/24         | RIP    | 100 | 1    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
|                    | RIP    | 100 | 1    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 255.255.255.255/32 | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |

The routing table shows that the next-hop address of the route destined for 172.16.1.0/24 is 2.2.2.2 and that the outbound interface is GigabitEthernet 1/0/0. This indicates that traffic is transmitted on the primary link DeviceA -> DeviceB.

### Step 3 Configure BFD for RIP.

# Configure BFD on all interfaces of DeviceA.

```
[~DeviceA] bfd
```

```
[*DeviceA-bfd] quit
[*DeviceA] rip 1
[*DeviceA-rip-1] bfd all-interfaces enable
[*DeviceA-rip-1] bfd all-interfaces min-rx-interval 100 min-tx-interval 100 detect-multiplier 10
[*DeviceA-rip-1] commit
[~DeviceA-rip-1] quit
```

The configuration on DeviceB is similar to the preceding configuration. For details, see configuration files.

# After completing the configurations, run the **display rip bfd session** command. The command output shows that a BFD session has been established between DeviceA and DeviceB and that the BFD State is Up. Use the command output on DeviceA as an example.

```
[~DeviceA] display rip 1 bfd session all
Interface :GigabitEthernet1/0/0
Locallp :2.2.2.1 Remotelp :2.2.2.2 BFDState :Up

Interface :GigabitEthernet2/0/0
Locallp :3.3.3.1 Remotelp :3.3.3.2 BFDState :Down
```

#### Step 4 Verifying the Configuration

# Run the **shutdown** command on GigabitEthernet 1/0/0 of DeviceB to simulate a fault on the primary link.



Link fault simulation is required for verification and does not need to be performed in actual applications.

```
[~DeviceB] interface gigabitethernet1/0/0
[~DeviceB-GigabitEthernet1/0/0] shutdown
[*DeviceB-GigabitEthernet1/0/0] commit
```

# Check information about the BFD session on DeviceA. The command output shows that no BFD session exists between DeviceA and DeviceB.

```
[~DeviceA] display rip 1 bfd session all
Interface :GigabitEthernet2/0/0
Locallp :3.3.3.1 Remotelp :3.3.3.2 BFDState :Down
```

# Check the routing table of DeviceA.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Destinations : 8 Routes : 8
Destination/Mask Proto Pre Cost Flags NextHop Interface
3.3.3.0/24 Direct 0 0 D 3.3.3.1 GigabitEthernet2/0/0
3.3.3.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
3.3.3.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
172.16.1.0/24 RIP 100 2 D 3.3.3.2 GigabitEthernet2/0/0
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

The routing table shows that the backup link DeviceA -> DeviceC -> DeviceB is used after the primary link fails. The next hop address of the route to 172.16.1.0/24 is 3.3.3.2, and the outbound interface is GigabitEthernet 2/0/0.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

bfd

interface gigabitethernet1/0/0
undo shutdown
ip address 2.2.2.1 255.255.255.0

interface gigabitethernet2/0/0
undo shutdown
ip address 3.3.3.1 255.255.255.0

rip 1
version 2
network 2.0.0.0
network 3.0.0.0
bfd all-interfaces enable
bfd all-interfaces min-tx-interval 100 min-rx-interval 100 detect-multiplier 10

return
```

- DeviceB configuration file

```

sysname DeviceB

bfd

interface gigabitethernet1/0/0
undo shutdown
ip address 2.2.2.2 255.255.255.0

interface gigabitethernet2/0/0
undo shutdown
ip address 4.4.4.1 255.255.255.0

interface gigabitethernet3/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
rip 1
version 2
network 2.0.0.0
network 4.0.0.0
network 172.16.0.0
bfd all-interfaces enable
bfd all-interfaces min-tx-interval 100 min-rx-interval 100 detect-multiplier 10

return
```

- DeviceC configuration file

```

sysname DeviceC

interface gigabitethernet1/0/0
undo shutdown
ip address 4.4.4.2 255.255.255.0

interface gigabitethernet2/0/0
undo shutdown
ip address 3.3.3.2 255.255.255.0

rip 1
version 2
network 3.0.0.0
network 4.0.0.0
#
```

```
return
● DeviceD configuration file
#
sysname DeviceD
#
interface gigabitethernet1/0/0
undo shutdown
ip address 172.16.1.2 255.255.255.0
#
rip 1
version 2
network 172.16.0.0
#
return
```

## Example for Configuring Static BFD for RIP

This section provides an example for configuring static BFD for RIP.

## Networking Requirements

RIP periodically exchanges Update messages to monitor the status of neighbors. By default, if a local device does not receive any Update messages from its neighbor after six update intervals (180s) elapse, it considers the neighbor down.

Voice, video, and other VOD services are widely used and are sensitive to the packet loss and delay. Long-time fault detection will cause the loss of a large number of data packets. As a result, the requirement of carrier-class networks for high reliability cannot be met. BFD for RIP can be deployed to complete link fault detection within milliseconds, speeding up RIP convergence.

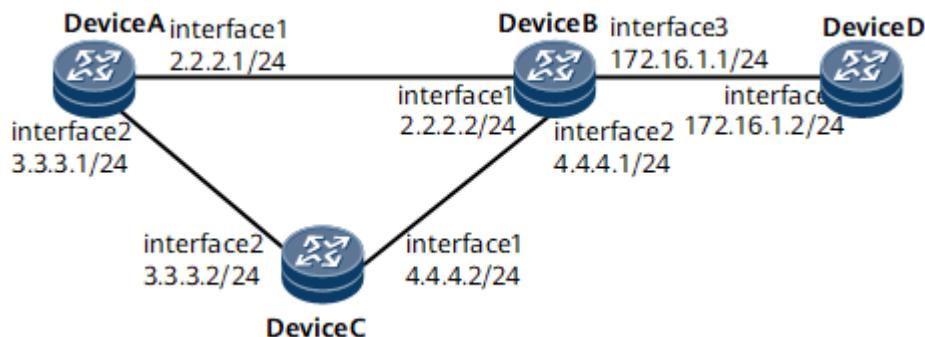
On the network shown in [Figure 1-215](#):

- DeviceA, DeviceB, DeviceC, and DeviceD run RIP.
- Traffic is transmitted along the primary link DeviceA -> DeviceB -> DeviceD. Static BFD is enabled on the interfaces connecting DeviceA and DeviceB. If the primary link fails, BFD can rapidly detect the fault and notify RIP of the fault, allowing service traffic to be rapidly switched to the backup link.

**Figure 1-215** Configuring static BFD for RIP



In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

To improve security, you are advised to configure RIP-2 packet authentication. For details, see "Improving RIP Network Security." The following example describes how to configure an authentication mode for RIP-2 packets. For details, see "Example for Configuring Basic RIP Functions."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic RIP functions on each router to establish RIP neighbor relationships.
2. Enable BFD globally.
3. Enable static BFD on the interfaces connecting DeviceA and DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

- DeviceA: RIP process ID (1), RIP version number (2), IP address of GE 1/0/0 (2.2.2.1/24), and IP address of GE 2/0/0 (3.3.3.1/24)
- DeviceB: RIP process ID (1), RIP version number (2), and IP addresses of GE 1/0/0, GE 2/0/0, and GE3/0/0 (2.2.2.2/24, 4.4.4.1/24, and 172.16.1.1/24, respectively)
- DeviceC: RIP process ID (1), RIP version number (2), IP address of GE 1/0/0 (4.4.4.2/24), and IP address of GE 2/0/0 (3.3.3.2/24)
- DeviceD: RIP process ID (1), RIP version number (2), and IP address of GE 1/0/0 (172.16.1.2/24)
- Local and remote discriminators of the BFD session established between DeviceA and DeviceB

## Procedure

### Step 1 Configure IP addresses for interfaces.

Assign an IP address to each interface according to [Figure 1-215](#) and [Data Preparation](#). For configuration details, see configuration files in this section.

### Step 2 Configure basic RIP functions.

# Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] rip 1
[*DeviceA-rip-1] version 2
[*DeviceA-rip-1] network 2.0.0.0
[*DeviceA-rip-1] network 3.0.0.0
[*DeviceA-rip-1] commit
[~DeviceA-rip-1] quit
```

# Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] rip 1
[*DeviceB-rip-1] version 2
[*DeviceB-rip-1] network 2.0.0.0
```

```
[*DeviceB-rip-1] network 4.0.0.0
[*DeviceB-rip-1] network 172.16.0.0
[*DeviceB-rip-1] commit
[~DeviceB-rip-1] quit
```

# Configure DeviceC.

```
<DeviceC> system-view
[~DeviceC] rip 1
[*DeviceC-rip-1] version 2
[*DeviceC-rip-1] network 3.0.0.0
[*DeviceC-rip-1] network 4.0.0.0
[*DeviceC-rip-1] commit
[~DeviceC-rip-1] quit
```

# Configure DeviceD.

```
<DeviceD> system-view
[~DeviceD] rip 1
[*DeviceD-rip-1] version 2
[*DeviceD-rip-1] network 172.16.0.0
[*DeviceD-rip-1] commit
[~DeviceD-rip-1] quit
```

# After completing the preceding configurations, run the **display rip neighbor** command. The command output shows that DeviceA has established neighbor relationships with DeviceB and DeviceC. Use the command output on DeviceA as an example.

```
[~DeviceA] display rip 1 neighbor
```

| IP Address              | Interface            | Type | Last-Heard-Time |
|-------------------------|----------------------|------|-----------------|
| 3.3.3.2                 | GigabitEthernet2/0/0 | RIP  | 0:0:5           |
| Number of RIP routes :2 |                      |      |                 |
| 2.2.2.2                 | GigabitEthernet1/0/0 | RIP  | 0:0:5           |
| Number of RIP routes :4 |                      |      |                 |

# Run the **display ip routing-table** command. The command output shows that routers can import routes from each other. Take the command output on DeviceA as an example.

```
[~DeviceA] display ip routing-table
```

```
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table : _public_
Destinations : 15 Routes : 15
```

| Destination/Mask   | Proto  | Pre | Cost | Flags | NextHop   | Interface            |
|--------------------|--------|-----|------|-------|-----------|----------------------|
| 3.0.0.0/8          | RIP    | 100 | 3    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
| 3.3.3.0/24         | Direct | 0   | 0    | D     | 3.3.3.1   | GigabitEthernet2/0/0 |
| 3.3.3.1/32         | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 3.3.3.255/32       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 2.0.0.0/8          | RIP    | 100 | 3    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 2.2.2.0/24         | Direct | 0   | 0    | D     | 2.2.2.1   | GigabitEthernet1/0/0 |
| 2.2.2.1/32         | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 2.2.2.255/32       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 127.0.0.0/8        | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32       | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.255.255.255/32 | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 172.16.0.0/16      | RIP    | 100 | 4    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 172.16.1.0/24      | RIP    | 100 | 1    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 4.4.4.0/24         | RIP    | 100 | 1    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
|                    | RIP    | 100 | 1    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 255.255.255.255/32 | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |

| Destination/Mask   | Proto  | Pre | Cost | Flags | NextHop   | Interface            |
|--------------------|--------|-----|------|-------|-----------|----------------------|
| 3.0.0.0/8          | RIP    | 100 | 3    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
| 3.3.3.0/24         | Direct | 0   | 0    | D     | 3.3.3.1   | GigabitEthernet2/0/0 |
| 3.3.3.1/32         | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 3.3.3.255/32       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 2.0.0.0/8          | RIP    | 100 | 3    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 2.2.2.0/24         | Direct | 0   | 0    | D     | 2.2.2.1   | GigabitEthernet1/0/0 |
| 2.2.2.1/32         | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 2.2.2.255/32       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet1/0/0 |
| 127.0.0.0/8        | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32       | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.255.255.255/32 | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 172.16.0.0/16      | RIP    | 100 | 4    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 172.16.1.0/24      | RIP    | 100 | 1    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 4.4.4.0/24         | RIP    | 100 | 1    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
|                    | RIP    | 100 | 1    | D     | 2.2.2.2   | GigabitEthernet1/0/0 |
| 255.255.255.255/32 | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |

The routing table shows that the next-hop address of the route destined for 172.16.0.0/16 is 2.2.2.2 and that the outbound interface is GigabitEthernet 1/0/0. This indicates that traffic is transmitted on the primary link DeviceA -> DeviceB.

**Step 3** Configure static BFD.

# Configure static BFD on DeviceA.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bfd 1 bind peer-ip 2.2.2.2 interface gigabitethernet1/0/0 source-ip 2.2.2.1
[*DeviceA-bfd-session-1] discriminator local 1
[*DeviceA-bfd-session-1] discriminator remote 2
[*DeviceA-bfd-session-1] commit
[~DeviceA-bfd-session-1] quit
[~DeviceA] interface gigabitethernet1/0/0
[~DeviceA-GigabitEthernet1/0/0] rip bfd static
```

# Configure static BFD on DeviceB.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] bfd 1 bind peer-ip 2.2.2.1 interface gigabitethernet1/0/0 source-ip 2.2.2.2
[*DeviceB-bfd-session-1] discriminator local 2
[*DeviceB-bfd-session-1] discriminator remote 1
[*DeviceB-bfd-session-1] commit
[~DeviceB-bfd-session-1] quit
[~DeviceB] interface gigabitethernet1/0/0
[~DeviceB-GigabitEthernet1/0/0] rip bfd static
```

# After completing the configurations, run the **display bfd session all** command on DeviceA. The command output shows that a static BFD session has been established.

```
[~DeviceA] display bfd session all
(w): State in WTR
(*) : State is invalid
```

| Local | Remote | PeerIpAddr | State | Type    | InterfaceName        |
|-------|--------|------------|-------|---------|----------------------|
| 1     | 2      | 2.2.2.2    | Up    | S_IP_IF | GigabitEthernet1/0/0 |

```
Total UP/DOWN Session Number : 1/0
```

**Step 4** Verifying the Configuration

# Run the **shutdown** command on GigabitEthernet 1/0/0 of DeviceB to simulate a fault on the primary link.

 **NOTE**

Link fault simulation is required for verification and does not need to be performed in actual applications.

```
[~DeviceB] interface gigabitethernet1/0/0
[~DeviceB-GigabitEthernet1/0/0] shutdown
[*DeviceB-GigabitEthernet1/0/0] commit
```

# Check information about the BFD session on DeviceA. The command output shows that no BFD session exists between DeviceA and DeviceB.

# Check the routing table of DeviceA.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
```

| Destinations : 8      Routes : 8 |        |     |      |       |           |                      |
|----------------------------------|--------|-----|------|-------|-----------|----------------------|
| Destination/Mask                 | Proto  | Pre | Cost | Flags | NextHop   | Interface            |
| 3.3.3.0/24                       | Direct | 0   | 0    | D     | 3.3.3.1   | GigabitEthernet2/0/0 |
| 3.3.3.1/32                       | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 3.3.3.255/32                     | Direct | 0   | 0    | D     | 127.0.0.1 | GigabitEthernet2/0/0 |
| 127.0.0.0/8                      | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.0.0.1/32                     | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 127.255.255.255/32               | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |
| 172.16.1.0/24                    | RIP    | 100 | 2    | D     | 3.3.3.2   | GigabitEthernet2/0/0 |
| 255.255.255.255/32               | Direct | 0   | 0    | D     | 127.0.0.1 | InLoopBack0          |

The routing table shows that the backup link DeviceA -> DeviceC -> DeviceB is used after the primary link fails. The next hop address of the route to 172.16.1.0/24 is 3.3.3.2, and the outbound interface is GigabitEthernet 2/0/0.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

bfd

interface gigabitethernet1/0/0
undo shutdown
ip address 2.2.2.1 255.255.255.0
rip bfd static

interface gigabitethernet2/0/0
undo shutdown
ip address 3.3.3.1 255.255.255.0

rip 1
version 2
network 2.0.0.0
network 3.0.0.0

bfd 1 bind peer-ip 2.2.2.2 interface gigabitethernet1/0/0 source-ip 2.2.2.1
discriminator local 1
discriminator remote 2

return
```

- DeviceB configuration file

```

sysname DeviceB

bfd

interface gigabitethernet1/0/0
undo shutdown
ip address 2.2.2.2 255.255.255.0
rip bfd static

interface gigabitethernet2/0/0
undo shutdown
ip address 4.4.4.1 255.255.255.0

interface gigabitethernet3/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0

rip 1
```

```
version 2
network 2.0.0.0
network 4.0.0.0
network 172.16.0.0
#
bfd 1 bind peer-ip 2.2.2.1 interface gigabitethernet1/0/0 source-ip 2.2.2.2
discriminator local 2
discriminator remote 1
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface gigabitethernet1/0/0
undo shutdown
ip address 4.4.4.2 255.255.255.0
#
interface gigabitethernet2/0/0
undo shutdown
ip address 3.3.3.2 255.255.255.0
#
rip 1
version 2
network 3.0.0.0
network 4.0.0.0
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
interface gigabitethernet1/0/0
undo shutdown
ip address 172.16.1.2 255.255.255.0
#
rip 1
version 2
network 172.16.0.0
#
return
```

## 1.1.7 RIPng Configuration

### 1.1.7.1 RIPng Description

#### 1.1.7.1.1 Overview of RIPng

##### Definition

RIP next generation (RIPng) is an extension to RIP version 2 (RIP-2) on IPv6 networks. Most RIP concepts apply to RIPng.

RIPng is a distance-vector routing protocol, which measures the distance (metric or cost) to the destination host by the hop count. In RIPng, the hop count from a device to its directly connected network is 0, and the hop count from a device to a network that is reachable through another device is 1. When the hop count is equal to or exceeds 16, the destination network or host is defined as unreachable.

To be applied on IPv6 networks, RIPng makes the following changes to RIP:

- UDP port number: RIPng uses UDP port number 521 to send and receive routing information.
- Multicast address: RIPng uses FF02::9 as the link-local multicast address of a RIPng device.
- Prefix length: RIPng uses a 128-bit (the mask length) prefix in the destination address.
- Next hop address: RIPng uses a 128-bit IPv6 address.
- Source address: RIPng uses link-local address FE80::/10 as the source address to send RIPng Update packets.

## Purpose

RIPng is an extension to RIP for support of IPv6.

### 1.1.7.1.2 Understanding RIPng

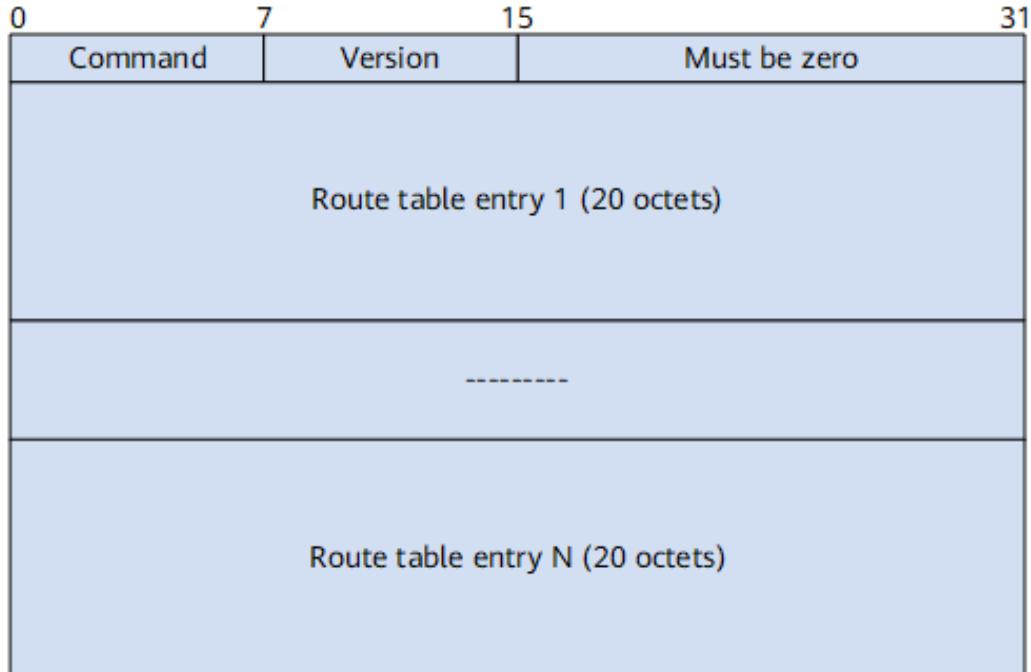
RIPng is an extension to RIPv2 on IPv6 networks and uses the same timers as RIPv2. RIPng supports split horizon, poison reverse, and triggered update, which prevents routing loops.

## RIPng Packet Format

A RIPng packet is composed of a header and multiple route table entries (RTEs). In a RIPng packet, the maximum number of RTEs is determined by the maximum transmission unit (MTU) of an interface.

[Figure 1-216](#) shows the basic format of a RIPng packet.

**Figure 1-216** RIPng packet format

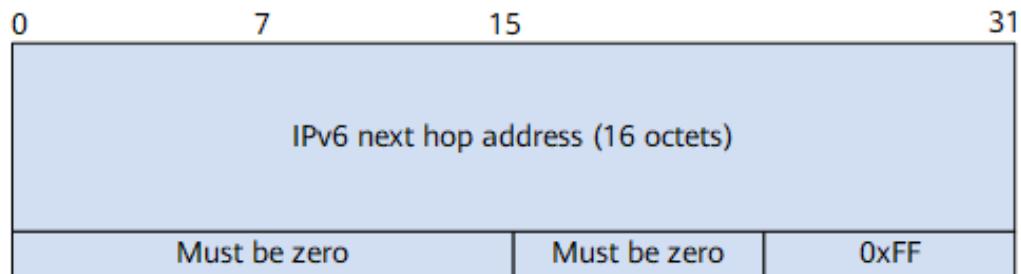


A RIPng packet contains two types of RTEs:

- Next hop RTE: It defines the IPv6 address of the next hop and is located before a group of IPv6-prefix RTEs that have the same next hop. The Metric field of a next hop RTE is always 0xFF.
- IPv6-prefix RTE: It describes the destination IPv6 address and the cost in the RIPng routing table and is located after a next hop RTE. A next hop RTE can be followed by multiple different IPv6-prefix RTEs.

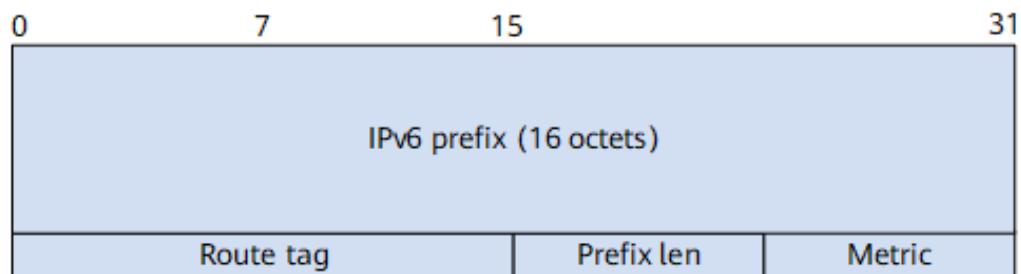
[Figure 1-217](#) shows the format of a next hop RTE.

**Figure 1-217** Format of the next hop RTE



[Figure 1-218](#) shows the format of an IPv6-prefix RTE.

**Figure 1-218** Format of the IPv6-prefix RTE



## Timers

RIPng uses the following timers:

- Update timer: This timer periodically triggers Update packet transmission. By default, the interval at which Update packets are sent is 30s. This timer is used to synchronize RIPng routes on the network.
- Age timer: If a RIPng device does not receive any Update packet from its neighbor before a route expires, the RIPng device considers the route to its neighbor unreachable.
- Garbage-collect timer: If no packet is received to update an unreachable route after the Age timer expires, this route is deleted from the RIPng routing table.

- Hold-down timer: If a RIP device receives an updated route with cost 16 from a neighbor, the route enters the holddown state, and the hold-down timer is started.

The following describes the relationship among these timers:

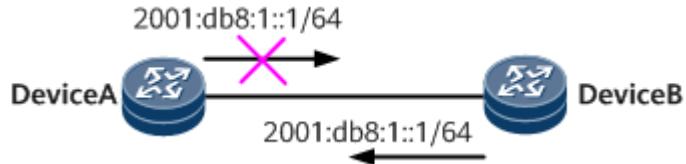
The advertisement of RIPng routing updates is periodically triggered by the update timer with default value 30 seconds. Each routing entry is associated with two timers: the Age timer and garbage-collect timer. Each time a route is learned and added to the routing table, the Age timer is started. If no Update packet is received from the neighbor within 180 seconds, the metric of the route is set to 16, and the garbage-collect timer is started. If no Update packet is received within 120 seconds, the route is deleted after the garbage-collect timer expires.

By default, hold-down timer is disabled. If you configure a hold-down timer, it starts after the system receives a route with a cost greater than 16 from its neighbor.

## Split Horizon

Split horizon prevents a RIPng-enabled interface from sending back the routes it learns, which reduces bandwidth consumption and prevents routing loops.

**Figure 1-219** Networking for split horizon



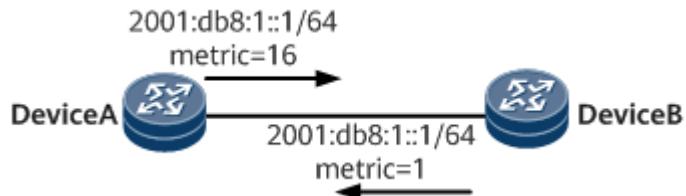
On the network shown in [Figure 1-219](#), after DeviceB sends a route to network 2001:db8:1::1 to DeviceA, DeviceA does not send the route back to DeviceB.

## Poison Reverse

Poison reverse allows a RIPng-enabled interface to set the cost of the route that it learns from a neighbor to 16 (indicating that the route is unreachable) and then send the route back. After receiving this route, the neighbor deletes the useless route from its routing table.

Poison reverse of RIPng can also prevent routing loops.

**Figure 1-220** Networking for poison reverse



In [Figure 1-220](#), if split horizon is not configured, DeviceB sends the route learned from DeviceA back. The cost of the route from DeviceA to network 2001:db8:1::1/64 is 1. If this route becomes unreachable and DeviceB does not

receive any Update packet from DeviceA, DeviceB keeps sending the route destined for 2001:db8:1::1/64 to DeviceA. As a result, a routing loop occurs.

If DeviceA sends a route unreachability message to DeviceB after receiving a route from DeviceB, DeviceB no longer learns the reachable route from DeviceA. This prevents a routing loop.

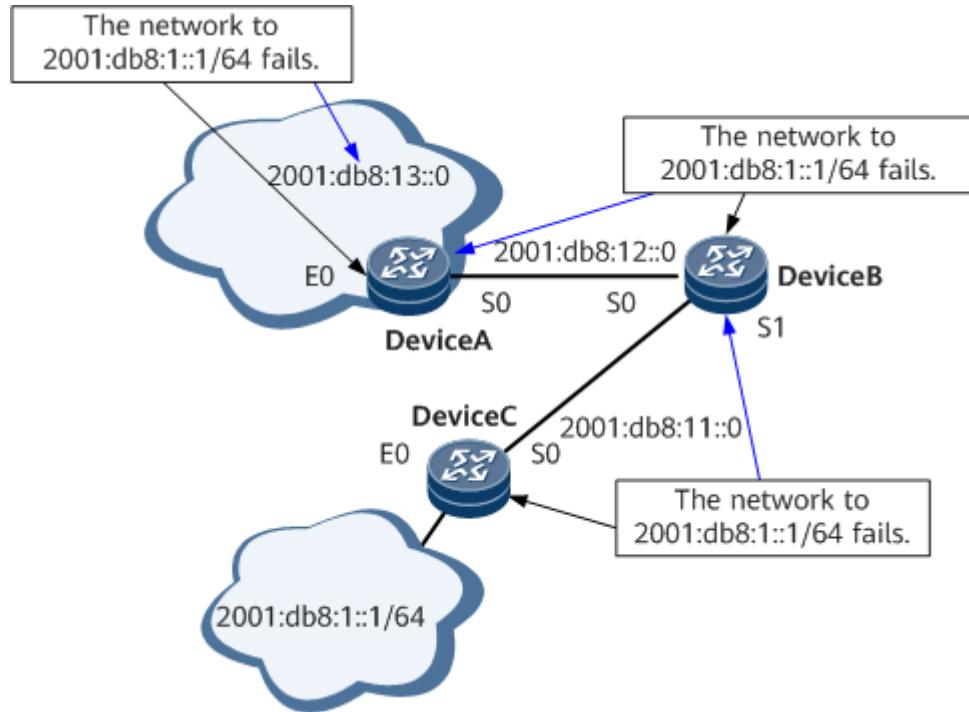
If both split horizon (preventing the routes learned by an interface from being sent back through this interface) and poison reverse are configured, only poison reverse takes effect.

## Triggered Update

Triggered update means that when routing information changes, a device immediately notifies its neighboring devices of the change through triggered update messages.

Triggered update allows a device to immediately broadcast each routing entry change to other devices without waiting for periodic update. This shortens the network convergence time.

**Figure 1-221** Networking for triggered update



In **Figure 1-221**, if network 2001:db8:1::1/64 becomes unreachable, DeviceC learns the information first. Generally, updated routing information is sent to neighboring Devices periodically. For example, RIPng sends the route update message every 30s. If the Update message from DeviceB reaches DeviceC when DeviceC is still waiting for the next round of periodic update, DeviceC learns the incorrect route to 2001:db8:1::1/64 from DeviceB. In this case, the next hops of the routes from DeviceB and DeviceC to network 2001:db8:1::1/64 are DeviceC and DeviceB, respectively. This results in a routing loop. If DeviceC sends an Update message to DeviceB immediately when detecting the network fault, DeviceB can update its routing table in time. This prevents the preceding problem.

Another method of implementing triggered update is as follows: When the next hop of a route becomes unavailable, due to a link failure for example, the local router sets the cost of the route to 16 and then advertises the route immediately to its neighbors. This process is called route poisoning.

## Route Summarization

### Context

On large-scale networks, the RIPng routing table of each device contains a large number of routing entries, which consumes lots of system resources. In addition, if a link within an IP address range frequently alternates between up and down, route flapping occurs.

RIPng route summarization was introduced to address these problems, enabling a device to summarize routes destined for different subnets of a network segment into one route destined for one network segment and then advertise the summary route to other network segments. RIPng route summarization reduces the number of routes in the routing table, minimizes system resource consumption, and prevents route flapping.

### Implementation

RIPng route summarization is implemented on interfaces. After RIPng route summarization is enabled on an interface, the interface summarizes routes based on the longest match rule and the metric of the summary route is the minimum metric among all the original routes.

For example, RIPng can advertise two routes, 2001:db8:1::1 (metric = 2) and 2001:db8:2::1 (metric = 3), through an interface. After route summarization is configured on the interface, the two routes are summarized into one route 2001:DB8::/32 (metric = 2), and this summary route is advertised.

### Multi-Process and Multi-Instance

RIPng supports multi-process and multi-instance, which simplifies network management and improves service control efficiency. Multi-process allows a set of interfaces to be associated with a specific RIPng process, which ensures that the specific RIPng process performs all the protocol operations only on this set of interfaces. Therefore, multiple RIPng processes can run on one router, and each process manages a unique set of interfaces. In addition, the routing data of each RIPng process is independent; however, processes can import routes from each other.

On routers that support VPN, each RIPng process is associated with a specific VPN instance. Therefore, all the interfaces associated with the RIPng process need to be associated with the RIPng process-related VPN instance.

### Hot Standby

Devices of a distributed architecture support RIPng Hot Standby (HSB). In the RIPng HSB process, RIPng backs up RIPng configuration from the Active Main Board (AMB) to the Standby Main Board (SMB). Whenever the AMB fails, the SMB becomes active. After the SMB is activated, RIPng resends a request to neighbors to synchronize the route database. Therefore, RIPng is not affected.

## IPsec Authentication

### Background

As networks develop, network security has become an increasing concern. Internet Protocol Security (IPsec) authentication can be used to authenticate RIPng packets. The packets that fail to be authenticated are discarded, which prevents data transmitted based on TCP/IP from being illegally obtained, tampered with, or attacked.

### Implementation

IPsec has an open standard architecture and ensures secure packet transmission on the Internet by encrypting packets. RIPng IPsec provides a complete set of security protection mechanisms to authenticate RIPng packets, which prevents devices from being attacked by forged RIPng packets.

IPsec includes a set of protocols that are used at the network layer to ensure data security, such as Internet Key Exchange (IKE), Authentication Header (AH), and Encapsulating Security Payload (ESP). The three protocols are described as follows:

- AH: A protocol that provides data origin authentication, data integrity check, and anti-replay protection. AH does not encrypt packets to be protected.
- ESP: A protocol that provides IP packet encryption and authentication mechanisms besides the functions provided by AH. The encryption and authentication mechanisms can be used together or independently.

#### NOTE

AH and ESP can be used together or independently.

### Benefits

RIPng IPsec offers the following benefits:

- Improves carriers' reputation and competitiveness by preventing services from being tampered with or attacked by unauthorized users.
- Ensures confidentiality and integrity of user packets.

## RIPng NSR

### Background

As networks develop, the demand for data, audio, and video services is growing, which imposes increasing requirements on IP network reliability. If an AMB/SMB switchover is performed on a device during a maintenance operation or a single point of failure occurs, routes may fail to converge, which may result in traffic loss or even a network breakdown. Non-stop routing (NSR) can address this problem and ensure uninterrupted forwarding of key services.

### Related Concepts

- High availability (HA): supports data backup between the AMB and SMB.
- Non-stop forwarding (NSF): enables a node to use the GR mechanism to ensure uninterrupted transmission during an AMB/SMB switchover.

- NSR: allows a standby control plane to take over traffic from an active control plane if the active control plane fails, preventing the control planes of neighbors from detecting the fault.
- AMB and SMB: run control plane processes.

## Implementation

With RIPng NSR, RIPng real-time data is synchronized between the AMB and SMB. After an AMB/SMB switchover is performed on a device, the SMB takes over services from the AMB, and neighbors are unaware of the local fault. After the switchover, the new AMB restores RIPng immediately based on the synchronized RIPng real-time data. Therefore, neighbors are unaware of the switchover as well. RIPng NSR requires synchronization of the following data:

- All configuration data, such as information about neighbors, timer parameters, and process configurations.
- Dynamic data, such as the interface parameters and state, and information about neighbors and the link state database (LSDB).

### NOTE

For details about NSR, see "Uninterruptible Service Technology" in *Feature Description - Reliability*.

## Usage Scenario

NSR minimizes the impact of control plane faults and prevents route flapping on networks that require high reliability.

## Benefits

NSR improves network reliability and ensures uninterrupted traffic forwarding.

### 1.1.7.2 RIPng Configuration

RIPng is an extension of RIP to support IPv6.

#### 1.1.7.2.1 Overview of RIPng

RIPng protocol can implement the interworking of small and medium-sized networks.

## Definition

RIP next generation (RIPng) is an extension to RIP version 2 (RIP-2) on IPv6 networks. Most RIP concepts apply to RIPng.

RIPng is a distance-vector routing protocol, which measures the distance (metric or cost) to the destination host by the hop count. In RIPng, the hop count from a device to its directly connected network is 0, and the hop count from a device to a network that is reachable through another device is 1. When the hop count is equal to or exceeds 16, the destination network or host is defined as unreachable.

To be applied on IPv6 networks, RIPng makes the following changes to RIP:

- UDP port number: RIPng uses UDP port number 521 to send and receive routing information.
- Multicast address: RIPng uses FF02::9 as the link-local multicast address of a RIPng device.
- Prefix length: RIPng uses a 128-bit (the mask length) prefix in the destination address.
- Next hop address: RIPng uses a 128-bit IPv6 address.
- Source address: RIPng uses link-local address FE80::/10 as the source address to send RIPng Update packets.

## Purpose

RIPng is an extension to RIP for support of IPv6.

### 1.1.7.2.2 Configuration Precautions for RIPng

## Feature Requirements

None

### 1.1.7.2.3 Configuring Basic RIPng Functions

To use RIPng features, configure basic RIPng functions, including creating RIPng processes and enabling RIPng on interfaces.

## Usage Scenario

While RIPng is simple and easy to use, it offers less powerful functions than OSPFv3 and IS-IS. As such, RIPng is mainly used on small-scale networks.

Configuring basic RIPng functions is a prerequisite for building RIPng networks.

## Pre-configuration Tasks

Before configuring basic RIPng functions, complete the following tasks:

- Configure a link layer protocol.
- Configure network-layer addresses for interfaces to ensure that neighboring devices are reachable at the network layer.
- Enable IPv6 in the system view.

## Creating a RIPng process

The creation of a RIPng processes is a precondition of all RIP configurations.

## Usage Scenario

Before running RIPng on a router, you need to create a RIPng process on the router.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ripng [ process-id ]**

The RIPng process is created, and the RIPng view is displayed.

RIPng supports multi-instance. To bind a RIPng process to a VPN instance, you can run the **ripng [ process-id ] vpn-instance vpn-instance-name** command.



If you run RIPng-related commands in the interface view before enabling RIPng, the configurations take effect only after RIPng is enabled.

### Step 3 (Optional) Run **description description**

A description is configured for the RIPng process.

### Step 4 Run **commit**

The configuration is submitted.

----End

## Enabling RIPng on an Interface

After an interface is associated with a RIPng process, routing information on this interface can be exchanged through RIPng.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

The interface is on the network side of the router. Specifically, the router is connected to other devices through this interface. To enable the router to learn the routes to the network segment where this interface resides, ensure that the link status of the interface is Up.

### Step 3 Run **ripng process-id enable**

RIPng is enabled on the specified interface.



- In the interface view, this command cannot be executed if IPv6 has not been enabled on the interface.
- If the router is connected to other devices through multiple interfaces, repeatedly perform Step 2 and Step 3.

### Step 4 Run **commit**

The configuration is submitted.

----End

### (Optional) Configuring the RIPng Priority

When multiple routing protocols are running on the same device, you can adjust the priority of RIPng for route selection.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run ripng [ process-id ]

The RIPng process is created and the RIPng view is displayed.

### Step 3 Run preference { preference | { route-policy route-policy-name | route-filter route-filter-name } } \*

The RIPng priority is set.

The **preference** command can be used with the routing policy to set the priority for the routes that meet the matching conditions.

After RIPng routes are delivered to the Routing Management (RM), if the RIPng priority changes, the RM updates the routing table.

### Step 4 Run commit

The configuration is submitted.

----End

### (Optional) Enabling Check on Zero Fields of RIPng Packets

Certain fields in RIPng packets must be 0, and these fields are called zero fields.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run ripng [ process-id ]

The RIPng process is created, and the RIPng view is displayed.

### Step 3 Run checkzero

The check on zero fields of RIPng packets is configured.

### Step 4 Run commit

The configuration is submitted.

----End

## Verifying the Basic RIPng Configuration

After configuring basic RIPng functions, verify the current operating status and routing information of RIPng.

### Prerequisites

Basic RIPng functions have been configured.

### Procedure

- Run the **display ripng [ process-id | vpn-instance vpn-instance-name ]** command to check the operating status and configuration of RIPng.
- Run the **display ripng process-id route [ destination-address destination-address [ mask-length ] ] [ interface interface-type interface-number [ neighbor-address neighbor-address ] ]** command to check RIPng routes.

----End

### 1.1.7.2.4 Controlling the Sending and Receiving of RIPng Packets

By controlling the sending and receiving of RIPng packets, you can optimize the RIPng performance.

### Applicable Environment

To meet the requirements of complex networking, accurately control the sending and receiving of RIPng packets.

### Pre-configuration Tasks

Before configuring a router to control the sending and receiving of RIPng packets, complete the following tasks:

- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.7.2.3 Configuring Basic RIPng Functions](#)

### Configuration Procedure

Perform one or more of the following configuration tasks (excluding "Checking the Configuration") as required.

### Disabling Receiving of RIPng Packets on an Interface

Disabling interfaces from receiving RIPng packets is a method of preventing routing loops.

### Context

When a device running RIPng is connected to a network running other routing protocols, you can run the **undo ripng input** command on the interface that connects the device to the network to prevent the interface from receiving useless packets from the network.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **undo ripng input**

The interface is disabled from receiving RIPng packets.

### Step 4 Run **commit**

The configuration is submitted.

----End

## Disabling Sending of RIPng Packets on an Interface

Disabling an interface from sending RIPng packets is a method of preventing routing loops.

## Context

When a device running RIPng is connected to a network running other routing protocols, you can run the **undo ripng output** command on the interface that connects the device to the network to prevent the interface from sending useless packets to the network.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **undo ripng output**

The interface is disabled from sending RIPng packets.

### Step 4 Run **commit**

The configuration is submitted.

----End

## Setting the Interval for Sending Update Packets and the Maximum Number of Packets Sent Each Time

By setting the interval for sending packets and the maximum number of packets to be sent each time, you can optimize the RIPng performance.

## Context

Perform the following steps on the RIPng router:

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **ripng pkt-transmit { interval interval | number packet-count } \***

The interval for sending RIPng Update packets and the maximum number of packets sent each time are set on the specified interface.

### Step 4 Run **commit**

The configuration is submitted.

----End

## Verifying the Configuration of Controlling the Sending and Receiving of RIPng Packets

After controlling the sending and receiving of RIPng packets, verify the routing information in the RIPng database.

## Prerequisites

Configurations have been performed to control the sending and receiving of RIPng packets.

## Procedure

- Run the **display ripng process-id database [ verbose ] [ destination-address destination-address [ mask-length ] ] [ interface interface-type interface-number [ neighbor-address neighbor-address ] ]** command to check the routing information in the RIPng database.

----End

### 1.1.7.2.5 Preventing Routing Loops

RIPng is a distance vector routing protocol. Because RIPng devices advertise their routing tables to their neighbors, routing loops may occur.

## Usage Scenario

RIPng prevents routing loops by using the following mechanisms:

- Counting to infinity: RIPng defines the cost of 16 as infinity. If the cost of a route reaches 16 due to a routing loop, this route is considered unreachable.

- Split horizon: Split horizon prevents a RIPng-enabled interface from sending back the routes it learns, which reduces bandwidth consumption and prevents routing loops.
- Poison reverse: Poison reverse allows a RIPng-enabled interface to set the cost of the route that it learns from a neighbor to 16 (indicating that the route is unreachable) and then send the route back. After receiving this route, the neighbor deletes the useless route from its routing table, which prevents loops.
- Suppression timers: Suppress timers can prevent routing loops and reduce the possibility that receiving incorrect routes results in incorrect routing information.

 NOTE

If both split horizon and poison reverse are configured, only poison reverse takes effect.

## Pre-configuration Tasks

Before configuring RIPng to prevent routing loops on the network, complete the following tasks:

- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.7.2.3 Configuring Basic RIPng Functions](#)

## Configuration Procedure

Perform one or more of the following configurations as required.

### Configuring Split Horizon

You can configure split horizon to prevent routing loops.

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 3** Run **ripng split-horizon**

Split horizon is enabled.

If both split horizon and poison reverse are configured, only poison reverse takes effect.

**Step 4** Run **commit**

The configuration is submitted.

**----End**

## Configuring Poison Reverse

You can configure poison reverse to prevent routing loops.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

If both split horizon and poison reverse are configured, only poison reverse takes effect.

#### Step 3 Run **ripng poison-reverse**

Poison reverse is enabled.

#### Step 4 Run **commit**

The configuration is submitted.

----End

## Configuring Suppression Timers

Suppression timers can prevent routing loops and reduce the possibility that receiving incorrect routes results in incorrect routing information.

### Context

When hop count of a route increases, a device starts suppression timers and accepts the Update packet of this route and updates the routing table until the suppression timers expire.

Suppression timers delays the addition of incorrect routes to the routing Table and slows down route convergence on the entire network as well. Therefore, exercise caution when configuring the suppression timers.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ripng process-id**

The RIPng process is created, and the RIPng view is displayed.

#### Step 3 Run **timers ripng update age suppress garbage-collect**

Suppression timers are set.

#### Step 4 Run **commit**

The configuration is submitted.

----End

## Follow-up Procedure

RIPng has four timers: *update*, *age*, *suppress*, and *garbage-collect*. The value of *update* is less than that of *age*, and the value of *suppress* is less than that of *garbage-collect*. Setting improper values for the timers affects RIP convergence speed and even causes route flapping on the network. For example, if the value of *update* is greater than that of *age*, a device cannot inform its neighbors of the change of RIP routes immediately.

For the configurations of *update*, *age*, *suppress*, and *garbage-collect*, see [1.1.7.2.8 Configuring RIPng Timers](#).

## Verifying the Configuration of Routing Loop Prevention

After configuring routing loop prevention, verify the running status of RIPng, information about interfaces, and RIPng routing information.

### Prerequisites

Routing loop prevention has been configured.

### Procedure

- Run the **display ripng [ process-id | vpn-instance vpn-instance-name ]** command to check the current operating status and configuration of RIPng.
- Run the **display ripng process-id route [ destination-address destination-address [ mask-length ] ] [ interface interface-type interface-number [ neighbor-address neighbor-address ] ]** command to check RIPng routes.
- Run the **display ripng process-id interface** command to check information about RIPng interfaces.

----End

### 1.1.7.2.6 Adjusting RIPng Route Selection

You can adjust RIPng route selection on a complicated network.

### Usage Scenario

To flexibly apply RIPng on a network and meet various requirements of users, you can change RIPng route selection by setting different parameters.

### Pre-configuration Tasks

Before adjusting RIPng route selection, complete the following tasks:

- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.7.2.3 Configuring Basic RIPng Functions](#)

## Configuration Procedure

Perform one or more of the following configurations as required.

### Configuring the Additional Metric on an Interface

The additional metric is the metric (hop count) to be added to the original metric of a RIPng route. You can set additional metrics for received RIPng routes and those to be sent.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **ripng metricin value**

The metric to be added to received routes is set.

### Step 4 Set the metric to be added to routes to be sent.

Run any of the following commands as required:

- Configure a basic ACL:

a. Run **ripng metricout { value | { acl6-number| acl6-name acl6-name } value1 } \***

b. Run **quit**

Return to the system view.

c. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

d. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \*** A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.

- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:
  - Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.
  - Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- Based on an IP prefix list:

Run **ripng metricout { value | ipv6-prefix ipv6-prefix-name value1 }** \*

#### Step 5 Run commit

The configuration is submitted.

----End

### Follow-up Procedure

Run the **ripng metricin** command to set an additional metric for received RIPng routes. This command affects the metric of each received route in the routing table.

Run the **ripng metricout** command to set an additional metric for advertised RIPng routes. However, the metric of the route in the routing table remains unchanged.

If the metric plus the additional metric exceeds 16, 16 is used as the metric.

### Setting the Maximum Number of Equal-Cost Routes

You can set the maximum number of equal-cost RIPng routes to adjust the number of routes that load-balance traffic.

### Procedure

#### Step 1 Run interface

The system view is displayed.

#### Step 2 Run **ripng [ process-id ]**

The RIPng process is created, and the RIPng view is displayed.

#### Step 3 Run **maximum load-balancing number**

The maximum number of equal-cost routes is set.

#### Step 4 Run **commit**

The configuration is submitted.

----End

### Verifying the Configuration of RIPng Route Selection

After adjusting RIPng route selection, verify the running status of RIPng, information about interfaces, and RIPng routing information.

#### Prerequisites

RIPng route selection has been adjusted.

#### Procedure

- Run the **display ripng [ process-id | vpn-instance vpn-instance-name ]** command to check the current operating status and configuration of RIPng.
- Run the **display ripng process-id route [ destination-address destination-address [ mask-length ] ] [ interface interface-type interface-number [ neighbor-address neighbor-address ] ]** command to check RIPng routes.
- Run the **display ripng process-id interface** command to check information about RIPng interfaces.

----End

#### 1.1.7.2.7 Controlling RIPng Routing Information

In most cases, multiple protocols run on the same network. Therefore, you need to control routing information of every protocol to meet different networking requirements.

#### Applicable Environment

To meet the requirements of complex networking, accurately control the sending and receiving of RIPng routing information.

#### Pre-configuration Tasks

Before configuring a router to control RIPng routing information, complete the following tasks:

- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.7.2.3 Configuring Basic RIPng Functions](#)

#### Configuration Procedure

Perform one or more of the following configurations as required.

## Configuring RIPng to Import External Routes

RIPng can import routing information from other RIPng processes or routing protocols to enrich the RIPng routing table.

### Context

On a large-scale network, multiple routing protocols may be configured for devices in different areas. In this situation, configure RIPng to import routes from other processes or routing protocols to implement inter-area communication.

If RIPng needs to advertise routing information of other RIPbg processes or other routing protocols (such as direct, static, RIPng, OSPFv3, IS-IS, or BGP), you can specify *protocol* to filter routing information. If *protocol* is not specified, routing information to be advertised can be filtered, including the imported routes and local RIPng routes (functioning as direct routes).

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ripng [ process-id ]**

The RIPng process is created and the RIPng view is displayed.

#### Step 3 (Optional) Run **default-cost cost**

The default cost is set for the imported routes.

If no cost is specified when external routes are imported, the default cost 0 is used.

#### Step 4 Run **import-route { static | direct | bgp [ permit-ibgp ] | unr | { isis | ospfv3 | ripng } [ process-id ] } [ [ cost cost | inherit-cost ] | { route-policy route-policy-name | route-filter route-filter-name } ] \***

External routes are imported.

#### Step 5 Run **commit**

The configuration is submitted.

----End

## Configuring RIPng to Filter the Received Routes

You can configure a router to selectively receive routes.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ripng [ process-id ]**

The RIPng view is displayed.

**Step 3** Run the following commands as required to filter the received routes based on different policies.

- To configure ACL-based filtering, run the **filter-policy { acl6-number| acl6-name acl6-name } import** command.
  - a. Run **quit**  
Return to the system view.
  - b. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**  
The ACL view is displayed.
  - c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ]**  
\* A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- To configure IP prefix list-based filtering, run the **filter-policy ipv6-prefix ipv6-prefix-name import** command.
- To configure route-policy-based filtering, run the **filter-policy route-policy route-policy-name import** command.

RIPng can filter received routes based on an IP prefix list. Only the routes that meet the matching rules are added to the RIPng routing table.

**Step 4** Run **commit**

The configuration is submitted.

----End

## Configuring RIPng to Filter the Routes to be Sent

You can configure RIPng to filter the routes to be sent.

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **ripng [ process-id ]**

The RIPng view is displayed.

**Step 3** Run the following commands as required to filter the routes to be sent based on different policies.

- To configure ACL-based filtering, run the **filter-policy { acl6-number | acl6-name acl6-name } export [ protocol [ process-id ] ]** command.
  - a. Run **quit**  
Return to the system view.
  - b. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**  
The ACL view is displayed.
  - c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ]**<sup>\*</sup> A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.

- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- To configure IP prefix list-based filtering, run the **filter-policy ipv6-prefix ipv6-prefix-name export [ protocol [ process-id ] ]** command.
- To configure a route-policy-based filtering, run the **filter-policy route-policy route-policy-name export [ protocol [ process-id ] ]** command.

RIPng can filter the routes to be sent based on an ACL6, route-policy, or an IPv6 prefix list. Only the routes that meet the match conditions are advertised to neighbors. If no protocol is specified in the command, all the routing information to be advertised will be filtered, including the imported routes and local RIPng routes (similar to direct routes).

#### Step 4 Run **commit**

The configuration is submitted.

----End

## Configuring RIPng to Advertise the Default Routes

There are two methods of advertising RIPng default routes. You can configure a router to advertise RIPng default routes as required. Alternatively, you can specify the cost of the default routes to be advertised.

## Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 3 Run **ripng default-route { only | originate } [ cost cost | tag tag ] \***

RIPng is configured to advertise default routes.

Specify either of the following parameters as required:

- **only**: advertises only IPv6 default routes (::/0) and suppress the advertisement of other routes.

- **originate**: advertises IPv6 default routes (::/0) without affecting the advertisement of other routes.

#### Step 4 Run **commit**

The configuration is submitted.

----End

### Follow-up Procedure

RIPng default routes are advertised in Update packets through the specified interface, regardless of whether these routes exist in the IPv6 routing table.

### Configuring RIPng Route Summarization

Configuring RIPng route summarization reduces the routing table size and prevents route flapping.

### Context

The RIPng routing table of a device on a medium or large RIPng network contains a large number of routes. Storing the routes consumes a large number of memory resources, and transmitting and processing these routes consume lots of network resources. To address this issue, configure RIPng route summarization.

With RIPng route summarization, a device summarizes routes destined for different subnets of a network segment into one route destined for one network segment and then advertises the summary route to other network segments. RIPng route summarization reduces the number of routes in a routing table and minimizes system resource consumption. In addition, if a specific link frequently alternates between Up and Down, the link status changes will not be advertised to devices that are located beyond the summarized route network segment, which prevents route flapping and improves network stability.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 3 Run **ripng summary-address ipv6-address prefix-length [ avoid-feedback ]**

RIPng route summarization is configured.



After RIPng route summarization is configured on a device, the local routing table still contains all specific routes before the summarization, while the routing tables of neighbors of the local device contain only the summarized route.

The cost of the summarized route is the smallest cost of the routes that have been summarized.

#### Step 4 Run commit

The configuration is committed.

----End

### Verifying the Configuration of RIPng Routing Information Control

After controlling RIPng routing information, verify all activated routes in the RIPng database.

### Prerequisites

Configurations have been performed to control RIPng routing information.

### Procedure

- Run the **display ripng process-id database [ verbose ] [ destination-address destination-address [ mask-length ] ] [ interface interface-type interface-number [ neighbor-address neighbor-address ] ]** command to check routes in the RIPng database.

----End

#### 1.1.7.2.8 Configuring RIPng Timers

There are four RIPng timers: Update, Age, Suppress, and Garbage-collect timers. You can adjust the RIPng convergence speed by changing the values of the RIPng timers.

### Usage Scenario

The value of *update* is less than that of *age*, and the value of *suppress* is less than that of *garbage-collect*. Setting improper values for the timers affects RIP convergence speed and even causes route flapping on the network. For example, if the value of *update* is greater than that of *age*, a device cannot inform its neighbors of the change of RIP routes immediately. Configuring Suppression timers can prevent routing loops. For details, see [Configuring Suppression Timers](#).

### Pre-configuration Tasks

Before configuring RIPng timers, complete the following tasks:

- Configure IPv6 addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [1.1.7.2.3 Configuring Basic RIPng Functions](#)

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run ripng [ process-id ]

A RIPng process is created, and the RIPng view is displayed.

**Step 3 Run timers ripng update age suppress garbage-collect**

RIPng timers are configured.

**Step 4 Run commit**

The configuration is submitted.

----End

## Checking the Configurations

Run the following commands to check the previous configuration.

- Run the **display ripng** command to view the values of RIPng timers.

### 1.1.7.2.9 Configuring IPsec Authentication for RIPng

By default, IP security (IPsec) authentication is not configured. Configuring authentication is recommended to ensure system security.

## Applicable Environment

As networks develop rapidly, network security has become a major concern. If IPsec authentication is configured on a RIPng network, the sent and received RIPng packets will be authenticated, and those cannot pass authentication will be discarded. This can improve the security of the RIPng network.

There are two methods of configuring IPsec authentication for RIPng:

- One method is to configure IPsec authentication in RIPng processes. If IPsec authentication is enabled in a RIPng process, this configuration takes effect on all interfaces in this RIPng process. This method is recommended if IPsec authentication needs to be applied to all interfaces in a RIPng process.
- The other method is to configure IPsec authentication on RIPng interfaces. This method is recommended if IPsec authentication needs to be applied only to some interfaces in a RIPng process.

## Pre-configuration Tasks

Before configuring IPsec authentication for RIPng, complete the following tasks:

- Configuring basic IPsec functions
- [Configuring Basic RIPng Functions](#)

## Configuring IPsec Authentication for a RIPng Process

Configuring IP security (IPsec) authentication in the RIPng view is one of the methods used to configure IPsec authentication for RIPng.

## Context

After IPsec authentication is configured in the RIPng view, all interfaces in this RIPng process perform IPsec authentication on RIPng packets received and to be sent.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ripng [ process-id ]**

The RIPng view is displayed.

### Step 3 Run **ipsec sa sa-name**

IPsec authentication is enabled, and the name of a security association (SA) is specified.

### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring IPsec Authentication on a RIPng Interface

Configuring IP security (IPsec) authentication in the interface view is the other method used to configure IPsec authentication for RIPng.

## Context

An SA configured on an RIPng interface is used to authenticate the packets sent and received by the interface.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **ripng ipsec sa sa-name**

IPsec authentication is enabled on the interface, and the name of an SA is specified.



The **ripng ipsec sa** command takes precedence over the **ipsec sa** command. If both commands are run in respective views and different SA names are specified, only the configuration of the **ripng ipsec sa** command takes effect.

### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of IPsec Authentication for RIPng

After IP security (IPsec) authentication for RIPng is configured, you can check the security association (SA) used in IPsec authentication and statistics on the RIPng packets that failed authentication.

### Prerequisites

After IPsec authentication is enabled in a RIPng process or on a RIPng interface, the configuration takes effect immediately. There is no need to restart the RIPng process.

### Procedure

- Run the **display ripng process-id interface [ interface-type interface-number ] [ verbose ]** command to check the SA used in IPsec authentication.
- Run the **display ripng process-id statistics interface { all | interface-type interface-number [ verbose | neighbor neighbor-ipv6-address ] }** command to check the number of RIPng packets that failed authentication.

----End

### 1.1.7.2.10 Maintaining RIPng

RIPng maintenance is implemented through debugging. Debugging, however, affects the system performance.

### Clearing RIPng

To clear RIPng information, run the reset commands in the user view.

### Context

#### NOTICE

RIPng information cannot be restored after you clear it. Therefore, exercise caution when running the reset command.

### Procedure

- Run the **reset ripng process-id statistics interface { all | interface-type interface-number [ neighbor neighbor-ipv6-address ] }** command to clear the statistics of the counter that is maintained by a particular RIPng process. This command helps to re-collect statistics during debugging.

----End

### Debugging RIPng

If the debugging of a module is enabled in the user view, the device can generate debugging information. Debugging information shows the contents of the packets received or sent by the debugged module.

## Context

### NOTICE

Debugging affects system performance. Therefore, after debugging is complete, run the **undo debugging all** command to disable debugging immediately.

If a RIPng fault occurs, you can run the **debugging** command in the user view to enable RIPng debugging. The debugging information helps you locate and analyze the fault.

## Procedure

- Run the **debugging ripng process-id [ error | event | job | backup | interface interface-type interface-number ] | packet [ interface-type interface-number [ neighbor neighbor-address ] ] | route acl-number ] | route-processing [ acl-number ] ]** command to enable debugging for the RIPng process.
- Run the **debugging ripng miscellaneous** command to debug RIPng component-level information.

----End

### 1.1.7.2.11 RIPng Configuration Examples

This section provides RIPng configuration examples.

### Example for Configuring Basic RIPng Functions

This section describes how to configure basic RIPng functions.

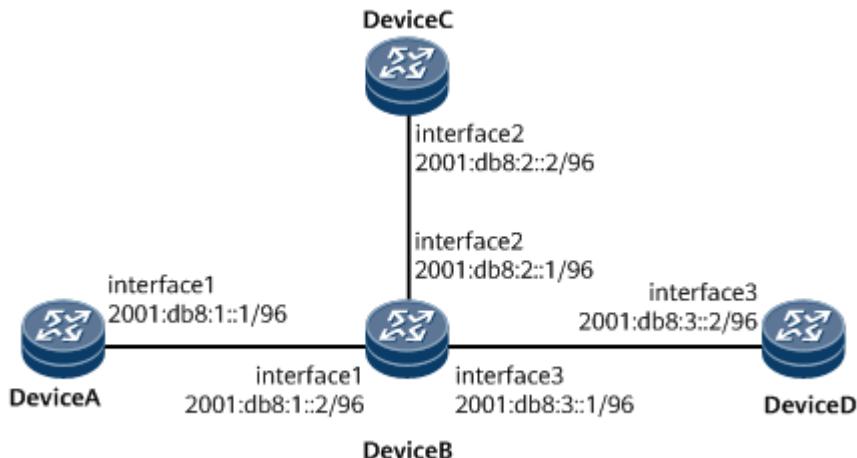
### Networking Requirements

On the network shown in [Figure 1-222](#), it is required that RIPng be enabled on all interfaces of DeviceA, DeviceB, DeviceC, and DeviceD and that these interfaces communicate with each other through RIPng.

**Figure 1-222** Network diagram of basic RIPng function configuration

### NOTE

In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

During the configuration, note the following:

- RIPng takes effect on an interface only after IPv6 is enabled.
- If a RIPng process is bound to a VPN instance, the interfaces running the RIPng process are also bound to the VPN instance.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Assign an IPv6 address to each interface to ensure network connectivity.
2. Enable RIPng and configure basic RIPng functions on each router.
3. Configure IPsec authentication for a RIPng process.

## Data Preparation

To complete the configuration, you need the following data:

- IPv6 address of the interface.

## Procedure

**Step 1** Assign an IPv6 address to each interface. For configuration details, see configuration files in this section.

**Step 2** Configure basic RIPng functions.

# Configure DeviceA.

```
[*DeviceA] ripng 1
[*DeviceA-ripng-1] quit
[*DeviceA] interface gigabitethernet1/0/0
[*DeviceA-GigabitEthernet1/0/0] ipv6 enable
[*DeviceA-GigabitEthernet1/0/0] ripng 1 enable
[*DeviceA-ripng-1] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[*DeviceB] ripng 1
[*DeviceB-ripng-1] quit
[*DeviceB] interface gigabitethernet1/0/0
[*DeviceB-GigabitEthernet1/0/0] ipv6 enable
[*DeviceB-GigabitEthernet1/0/0] ripng 1 enable
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet2/0/0
[*DeviceB-GigabitEthernet2/0/0] ipv6 enable
[*DeviceB-GigabitEthernet2/0/0] ripng 1 enable
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface gigabitethernet3/0/0
[*DeviceB-GigabitEthernet3/0/0] ipv6 enable
[*DeviceB-GigabitEthernet3/0/0] ripng 1 enable
[*DeviceB-GigabitEthernet3/0/0] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
[*DeviceC] ripng 1
[*DeviceC-ripng-1] quit
[*DeviceC] interface gigabitethernet2/0/0
[*DeviceC-GigabitEthernet2/0/0] ipv6 enable
[*DeviceC-GigabitEthernet2/0/0] ripng 1 enable
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[*DeviceD] ripng 1
[*DeviceD-ripng-1] quit
[*DeviceD] interface gigabitethernet3/0/0
[*DeviceD-GigabitEthernet3/0/0] ipv6 enable
[*DeviceD-GigabitEthernet3/0/0] ripng 1 enable
[*DeviceD-GigabitEthernet3/0/0] quit
[*DeviceD] commit
```

### Step 3 Configure IPsec authentication for a RIPng process.

# Create an IPSec proposal on DeviceA.

```
[~DeviceA] ipsec proposal proposal1
[*DeviceA-ipsec-proposal-proposal1] encapsulation-mode transport
[*DeviceA-ipsec-proposal-proposal1] transform esp
[*DeviceA-ipsec-proposal-proposal1] esp authentication-algorithm sha2-256
[*DeviceA-ipsec-proposal-proposal1] commit
[~DeviceA-ipsec-proposal-proposal1] quit
```

# Configure an IPsec SA and apply the IPsec proposal to the SA on DeviceA.

```
[~DeviceA] ipsec sa sa1
[*DeviceA-ipsec-sa-sa1] proposal proposal1
[*DeviceA-ipsec-sa-sa1] commit
```

# Configure an SPI and a key in the string format on DeviceA.

```
[~DeviceA] ipsec sa sa1
[*DeviceA-ipsec-sa-sa1] sa spi inbound esp 12345
[*DeviceA-ipsec-sa-sa1] sa spi outbound esp 12345
[*DeviceA-ipsec-sa-sa1] sa string-key inbound esp abcdef
[*DeviceA-ipsec-sa-sa1] sa string-key outbound esp abcdef
[*DeviceA-ipsec-sa-sa1] commit
[~DeviceA-ipsec-sa-sa1] quit
```

# Configure an SA in the RIPng process on DeviceA.

```
[~DeviceA] ripng 1
[*DeviceA-ripng-1] ipsec sa sa1
[*DeviceA-ripng-1] commit
```

The configurations of other devices are similar to the configuration of DeviceA. For configuration details, see configuration files.

**Step 4** Verify the configuration.

# Check the neighbors of DeviceA.

```
[~DeviceA] display ripng 1 neighbor
Neighbor : FE80::A0A:201:1 GigabitEthernet1/0/0
Protocol : RIPNG
```

The command output shows that DeviceA has established neighbor relationships with other devices on the network.

# Check the routing information of DeviceB.

```
[~DeviceB] display ripng 1 route
Route Flags: A - Aging, S - Suppressed, G - Garbage-collect

Peer FE80::F54C:0:9FDB:1 on GigabitEthernet1/0/0
Dest 2001:DB8:1::1/96,
 via FE80::F54C:0:9FDB:1, cost 1, tag 0, A, 3 Sec
Peer FE80::D472:0:3C23:1 on GigabitEthernet2/0/0
Dest 2001:DB8:2::2/96,
 via FE80::D472:0:3C23:1, cost 1, tag 0, A, 4 Sec
Peer FE80::D472:0:3C23:1 on GigabitEthernet3/0/0
Dest 2001:DB8:3::2/96,
 via FE80::D472:0:3C23:1, cost 1, tag 0, A, 4 Sec
```

The command output shows that DeviceB has learned routing information on the network.

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/96
ripng 1 enable
#
interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:8::1/128
ripng 1 enable
#
ipsec proposal proposal1
encapsulation-mode transport
esp authentication-algorithm sha2-256
esp encryption-algorithm aes-gcm-128
#
ipsec sa sa1
proposal proposal1
sa spi inbound esp 12345
sa string-key inbound esp %##%#<}{br9\zi%X+/Y@:Y>Lw(L\v##^KsM"/8RaRe$%##%
sa spi outbound esp 12345
sa string-key outbound esp %##%#<}{@X4355SE9JZTD0>GQf"}w2@X,k6.E\Z,z\{##%#%
#
ripng 1
ipsec sa sa1
#
return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/96
ripng 1 enable

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/96
ripng 1 enable

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/96
ripng 1 enable

interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:9::1/128
ripng 1 enable

ipsec proposal proposal1
encapsulation-mode transport
esp authentication-algorithm sha2-256
esp encryption-algorithm aes-gcm-128

ipsec sa sa1
proposal proposal1
sa spi inbound esp 12345
sa string-key inbound esp %##%#<}jb{br9\zi%X+/Y@:Y>Lw(L\v#^KsM"/8RaRe$%#%#
sa spi outbound esp 12345
sa string-key outbound esp %##%#<}j/@X4355SE9JZTD0>GQf"}w2@X,k6.E\Z,z\{##%#%#

ripng 1
ipsec sa sa1

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/96
ripng 1 enable

interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:7::1/128
ripng 1 enable

ipsec proposal proposal1
encapsulation-mode transport
esp authentication-algorithm sha2-256
esp encryption-algorithm aes-gcm-128

ipsec sa sa1
proposal proposal1
sa spi inbound esp 12345
sa string-key inbound esp %##%#<}jb{br9\zi%X+/Y@:Y>Lw(L\v#^KsM"/8RaRe$%#%#
sa spi outbound esp 12345
```

```
sa string-key outbound esp %#%#<}j/@X4355SE9JZTD0>GQf"}w2@X,k6.E\Z,z\{#%#%#
#
ripng 1
ipsec sa sa1
#
return
```

- DeviceD configuration file

```

sysname DeviceD

ipv6

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::2/96
ripng 1 enable
#
interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:6::1/128
ripng 1 enable
#
ipsec proposal proposal1
encapsulation-mode transport
esp authentication-algorithm sha2-256
esp encryption-algorithm aes-gcm-128
#
ipsec sa sa1
proposal proposal1
sa spi inbound esp 12345
sa string-key inbound esp %#%#<}jb{br9\zi%X+/Y@:Y>Lw(L\v#*^KsM"/8RaRe$%#%#
sa spi outbound esp 12345
sa string-key outbound esp %#%#<}j/@X4355SE9JZTD0>GQf"}w2@X,k6.E\Z,z\{#%#%#
#
ripng 1
ipsec sa sa1
#
return
```

## Example for Configuring Split Horizon

This section describes how to configure split horizon to prevent routing loops.

### Networking Requirements

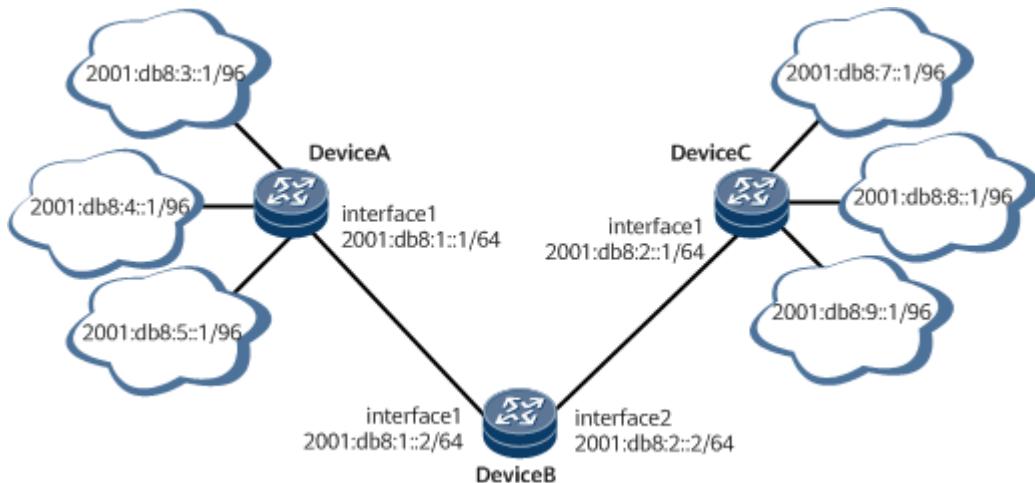
On the network shown in [Figure 1-223](#), IP addresses have been configured for interfaces on all routers, RIPng has been configured on each router, and RIPng services are running properly. Split horizon needs to be re-configured on DeviceA and DeviceC.

**Figure 1-223** Network diagram of split horizon



NOTE

Interface 1 and Interface 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



## Precautions

During the configuration, note the following:

When the split horizon and poison reverse are configured together, only poison reverse takes effect.

To improve security, you are advised to deploy IPsec authentication for RIPng. For details, see Configuring IPsec Authentication for RIPng. The following uses IPsec authentication in a RIPng process as an example. For details, see Example for Configuring Basic RIPng Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable split horizon.

## Data Preparation

To complete the configuration, you need the following data:

- RIPng network segments of DeviceA: 2001:db8:3::1/96, 2001:db8:4::1/96, and 2001:db8:5::1/96
- RIPng network segments of DeviceB: 2001:db8:1::2/64 and 2001:db8:2::2/64
- RIPng network segments of DeviceC: 2001:db8:7::1/96, 2001:db8:8::1/96, and 2001:db8:9::1/96
- IPv6 address of each interface

## Procedure

### Step 1 Configure split horizon.

Configure split horizon on the RIPng interfaces of all routers. The configuration of DeviceC is the same as the configuration of DeviceA and DeviceB, and is not mentioned here.

```
Configure DeviceA.
```

```
[~DeviceA] interface gigabitethernet1/0/0
[~DeviceA-GigabitEthernet1/0/0] ripng split-horizon
[*DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] interface gigabitethernet1/0/1
[~DeviceA-GigabitEthernet1/0/1] ripng split-horizon
[*DeviceA-GigabitEthernet1/0/1] quit
[~DeviceA] interface gigabitethernet1/0/2
[~DeviceA-GigabitEthernet1/0/2] ripng split-horizon
[*DeviceA-GigabitEthernet1/0/2] quit
[~DeviceA] interface gigabitethernet1/0/3
[~DeviceA-GigabitEthernet1/0/3] ripng split-horizon
[*DeviceA-GigabitEthernet1/0/3] quit
[*DeviceA] commit
```

**Step 2** Verify the configuration.

# Run the **display ripng 1 interface verbose** command on DeviceA and DeviceC to check the split horizon configuration. Use the command output on DeviceA as an example. If the displayed Split-Horizon field is Enabled, split horizon has been enabled.

```
[~DeviceA] display ripng 1 interface verbose
GigabitEthernet1/0/0
 FE80::A0A:200:1
 State : UP, Protocol : RIPNG, MTU : 1440
 Metricin : 0 , Metricout : 1
 Default Route : Disabled
 Poison Reverse : Disabled
 Split Horizon : Enabled
GigabitEthernet1/0/1(10.1.1.1)
 FE80::A0A:200:1
 State : UP, Protocol : RIPNG, MTU : 1440
 Metricin : 0 , Metricout : 1
 Default Route : Disabled
 Poison Reverse : Disabled
 Split Horizon : Enabled
GigabitEthernet1/0/2
 FE80::A0A:200:1
 State : UP, Protocol : RIPNG, MTU : 1440
 Metricin : 0 , Metricout : 1
 Default Route : Disabled
 Poison Reverse : Disabled
 Split Horizon : Enabled
GigabitEthernet1/0/3
 FE80::A0A:200:1
 State : UP, Protocol : RIPNG, MTU : 1440
 Metricin : 0 , Metricout : 1
 Default Route : Disabled
 Poison Reverse : Disabled
 Split Horizon : Enabled
```

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
ripng 1 enable
#
interface GigabitEthernet1/0/1
undo shutdown
```

```
ipv6 enable
ipv6 address 2001:db8:3::1/96
ripng 1 enable
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.2.1.1 255.255.0.0
#
interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:5::1/96
ripng 1 enable
#
ripng 1
#
return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
ripng 1 enable
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
ripng 1 enable
#
return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
ripng 1 enable
#
interface GigabitEthernet1/0/1
undo shutdown
ipv6 enable
ipv6 address 2001:db8:7::1/96
ripng 1 enable
#
interface GigabitEthernet1/0/2
undo shutdown
ipv6 enable
ipv6 address 2001:db8:8::1/96
ripng 1 enable
#
interface GigabitEthernet1/0/3
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9::1/96
ripng 1 enable
#
ripng 1
#
return
```

## 1.1.8 IS-IS Configuration

### 1.1.8.1 IS-IS Description

#### 1.1.8.1.1 Overview of IS-IS

##### Definition

Intermediate System to Intermediate System (IS-IS) is a dynamic routing protocol initially designed by the International Organization for Standardization (ISO) for the Connectionless Network Protocol (CLNP).

To support IP routing, the IETF extends and modifies IS-IS in relevant standards, which enables IS-IS to be applied to both TCP/IP and Open System Interconnection (OSI) environments. The new type of IS-IS is called integrated IS-IS or dual IS-IS.

In this document, IS-IS refers to integrated IS-IS, unless otherwise stated.



Unless otherwise specified, IS-IS features that support both IPv4 and IPv6 are implemented in the same way.

##### Purpose

IS-IS is an Interior Gateway Protocol (IGP) and is used within an autonomous system (AS). IS-IS is a link state protocol, and it uses the shortest path first (SPF) algorithm to calculate routes.

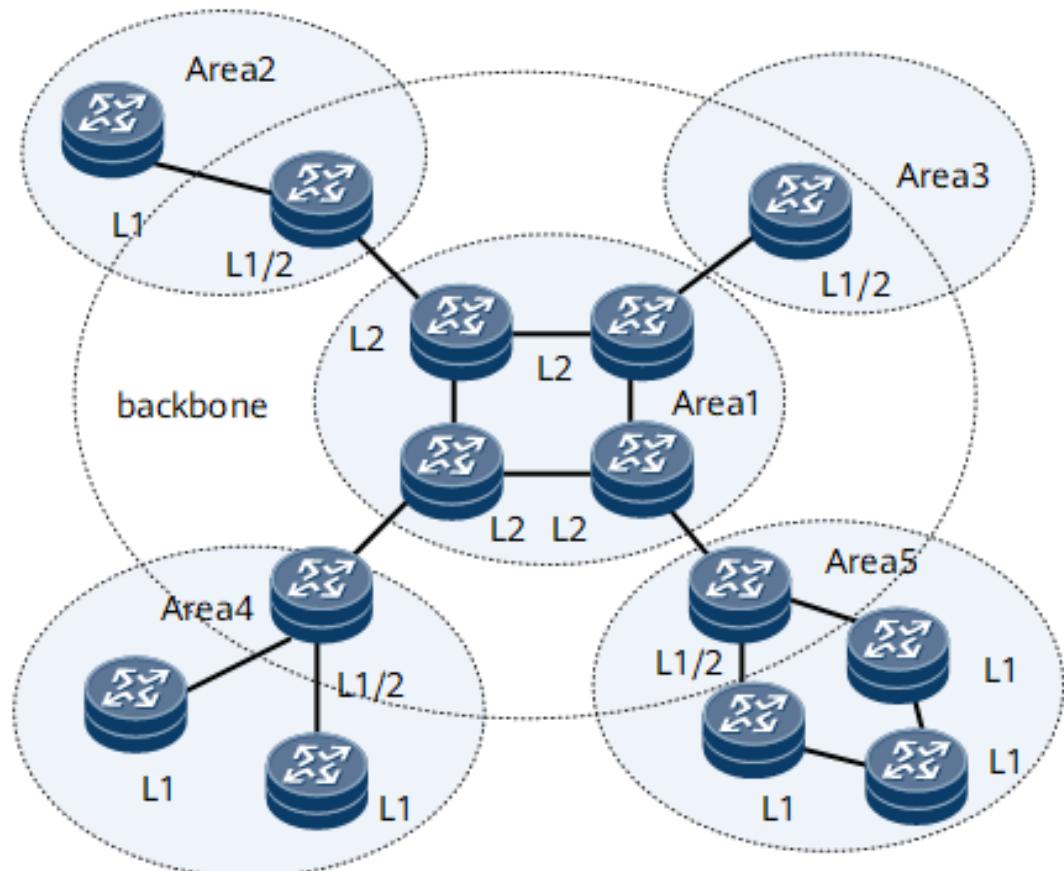
#### 1.1.8.1.2 Understanding IS-IS

##### Basic Concepts of IS-IS

##### IS-IS Areas

To support large-scale routing networks, IS-IS adopts a two-level structure in a routing domain. A large domain can be divided into areas. [Figure 1-224](#) shows an IS-IS network. The entire backbone area covers all Level-2 routers in area 1 and Level-1-2 routers in other areas. Three types of routers on the IS-IS network are described as follows:

Figure 1-224 IS-IS topology



- Level-1 router

A Level-1 router manages intra-area routing. It establishes Level-1 neighbor relationships only with other Level-1 devices and Level-1-2 routers in the same area, and maintains only a Level-1 link state database (LSDB) which contains routing information specific to the local area. For packets destined for other areas, each Level-1 router forwards them to the nearest Level-1-2 router.

- Level-2 router

A Level-2 router manages inter-area routing. It can establish Level-2 neighbor relationships with other Level-2 routers and with Level-1-2 routing devices in other areas and maintains a Level-2 LSDB, which contains inter-area routing information.

All Level-2 routers form the backbone network of the routing domain and are responsible for inter-area communication. The Level-2 routers in the routing domain must be contiguous to ensure backbone network continuity. Only Level-2 routers can directly exchange data packets or routing information with the routers beyond the area.

- Level-1-2 router

A router that belongs to both Level-1 and Level-2 areas is called a Level-1-2 router. It can establish Level-1 neighbor relationships with Level-1 routers in the same area, Level-2 neighbor relationships with Level-2 routers in other

areas, and Level-1 and Level-2 neighbor relationships with Level-1-2 routers. Level-1 routers can be connected to other areas only through Level-1-2 routers.

A Level-1-2 router maintains two LSDBs: a Level-1 LSDB and a Level-2 LSDB. The Level-1 LSDB is used for intra-area routing, whereas the Level-2 LSDB is used for inter-area routing.

#### NOTE

Level-1 routers in different areas cannot establish neighbor relationships. Level-1-2 routers can establish neighbor relationships with each other, regardless of the areas to which they belong.

Generally, Level-1 routers are deployed within an area, Level-2 routers are deployed between areas, and Level-1-2 routers are deployed between Level-1 and Level-2 routers.

#### Interface level

A Level-1-2 router may need to establish only a Level-1 adjacency with a peer and establish only a Level-2 adjacency with another peer. In this case, you can set the level of an interface to control the setting of adjacencies on the interface. Specifically, only Level-1 adjacencies can be established on a Level-1 interface, and only Level-2 adjacencies can be established on a Level-2 interface.

## Address Structure of IS-IS

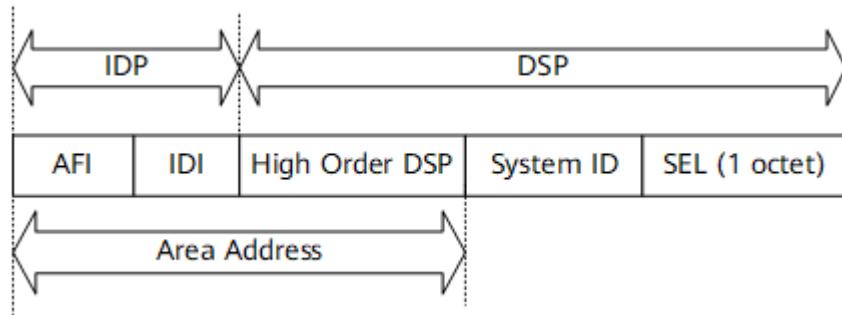
In OSI, the NSAP is used to locate resources. The ISO adopts the address structure shown in [Figure 1-225](#). An NSAP is composed of the Initial Domain Part (IDP) and the Domain Specific Part (DSP). IDP is the counterpart of network ID in an IP address, and DSP is the counterpart of the subnet number and host address in an IP address.

As defined by the ISO, the IDP consists of the Authority and Format Identifier (AFI) and Initial Domain Identifier (IDI). AFI specifies the address assignment mechanism and the address format; the IDI identifies a domain.

The DSP consists of the High Order DSP (HODSP), system ID, and NSAP Selector (SEL). The HODSP is used to divide areas; the system ID identifies a host; the SEL indicates the service type.

The lengths of the IDP and DSP are variable. The length of the NSAP varies from 8 bytes to 20 bytes.

**Figure 1-225** IS-IS address structure



- Area address

An IDP and HODSP of the DSP can identify a routing domain and the areas in a routing domain; therefore, the combination of the IDP and HODSP is referred to as an area address, equal to an area ID in OSPF. You are advised to avoid the situation where different Level-1 areas in the same routing domain have the same area address. The area addresses of routers in the same Level-1 area must be the same.

A router generally requires only one area address, and the area addresses of all nodes in the same area must be the same. In the implementation of a device, an IS-IS process can be configured with a maximum of three area addresses to support seamless combination, division, and transformation of areas.

- System ID

A system ID uniquely identifies a host or a router in an area. In the device, the length of the system ID is 48 bits (6 bytes).

A router ID corresponds to a system ID. If a router uses the IP address (192.168.1.1) of Loopback 0 as its router ID, its system ID used in IS-IS can be obtained through the following steps:

- Extend each part of the IP address 192.168.1.1 to 3 digits and add 0 or 0s to the front of the part that is shorter than 3 digits.
- Divide the extended address 192.168.001.001 into three parts, with each part consisting of 4 decimal digits.
- The reconstructed 1921.6800.1001 is the system ID.

There are many ways to specify a system ID. Whichever you choose, ensure that the system ID uniquely identifies a host or a router.

 NOTE

If the same system ID is configured for more than one device on the same network, network flapping may occur. To address this problem, IS-IS provides the automatic recovery function. With the function, if the system detects an IS-IS system ID conflict, it automatically changes the local system ID to resolve the conflict. The first two bytes of the system ID automatically changed by the system are Fs, and the last four bytes are randomly generated. For example, FFFF:1234:5678 is such a system ID. If the conflict persists after the system automatically changes three system IDs, the system no longer resolves this conflict.

- SEL

The role of an SEL (also referred to as NSAP Selector or N-SEL) is similar to that of the "protocol identifier" of IP. A transport protocol matches an SEL. The SEL is "00" in IP.

- NET

A Network Entity Title (NET) indicates the network layer information of an IS itself and consists of an area ID and a system ID. It does not contain the transport layer information (SEL = 0). A NET can be regarded as a special NSAP. The length of the NET field is the same as that of an NSAP, varying from 8 bytes to 20 bytes. For example, in NET **ab.cdef.1234.5678.9abc.00**, the area is **ab.cdef**, the system ID is **1234.5678.9abc**, and the SEL is **00**.

In general, an IS-IS process is configured with only one NET. When areas need to be redefined, for example, areas need to be combined or an area needs to be divided into sub-areas, you can configure multiple NETs.

#### NOTE

A maximum of three area addresses can be configured in an IS-IS process, and therefore, you can configure only a maximum of three NETs. When you configure multiple NETs, ensure that their system IDs are the same.

The routers in an area must have the same area address.

## IS-IS Network Types

IS-IS supports the following types of networks:

- Broadcast network
- Point-to-point (P2P) network

## Basic Protocols of IS-IS

### Related Concepts

#### DIS and Pseudo Node

A Designated Intermediate System (DIS) is an intermediate router elected in IS-IS communication. A pseudo node simulates a virtual node on a broadcast network and is not a real router. In IS-IS, a pseudo node is identified by the system ID and 1-byte circuit ID (a non-zero value) of a DIS.

The DIS is used to create and update pseudo nodes and generate the [link state protocol data units \(LSPs\)](#) of pseudo nodes. The routers advertise a single link to a pseudo node and obtain routing information about the entire network through the pseudo node. The router does not need to exchange packets with all the other routers on the network. Using the DIS and pseudo nodes simplifies network topology and reduces the length of LSPs generated by routers. When the network changes, fewer LSPs are generated. Therefore, fewer resources are consumed.

#### SPF Algorithm

The SPF algorithm, also named Dijkstra's algorithm, is used in a link-state routing protocol to calculate the shortest paths to other nodes on a network. In the SPF algorithm, a local router takes itself as the root and generates a shortest path tree (SPT) based on the network topology to calculate the shortest path to every destination node on a network. In IS-IS, the SPF algorithm runs separately in Level-1 and Level-2 databases.

## Implementation

All routers on the IS-IS network communicate through the following steps:

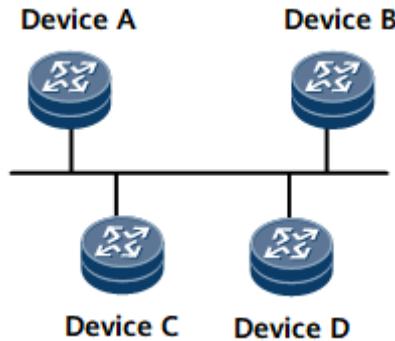
- [Establishment of IS-IS Neighbor Relationships](#)
- [LSDB Synchronization](#)
- [Route Calculation](#)

#### Establishment of IS-IS Neighbor Relationships

On different types of networks, the modes for establishing IS-IS neighbor relationships are different.

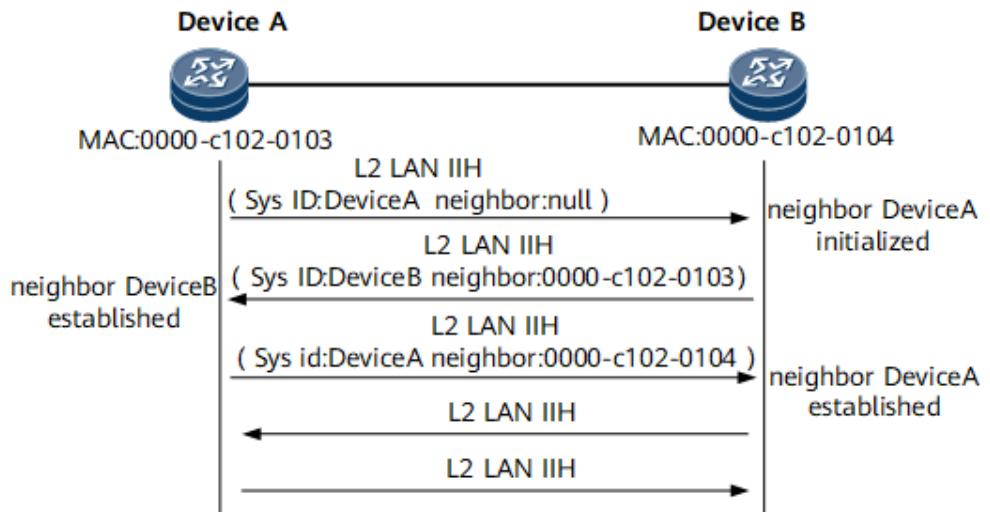
- Establishment of a neighbor relationship on a broadcast link

Figure 1-226 Networking for a broadcast link



Device A, Device B, Device C, and Device D are Level-2 routers. Device A is newly added to the broadcast network. [Figure 1-227](#) demonstrates the process of establishing the neighbor relationship between Device A and Device B, the process of establishing the neighbor relationship between Device A and Device C or Device D is similar to that between Device A and Device B.

Figure 1-227 Establishing a neighbor relationship on a broadcast link



As shown in [Figure 1-227](#), the process for establishing a neighbor relationship on a broadcast link consists of the following phases:

- Device A broadcasts a Level-2 local area network (LAN) **IS-to-IS Hello PDU (IIH)**. After Device B receives the IIH, Device B detects that the neighbor field in the IIH does not contain its media access control (MAC) address, and sets its neighbor status with Device A to **Initial**.
- Device B replies a Level-2 LAN IIH to Device A. After Device A receives the IIH, Device A detects that the neighbor field in the IIH contains its MAC address, and sets its neighbor status with Device B to **Up**.

- Device A sends a Level-2 LAN IIH to Device B. After Device B receives the IIH, Device B detects that the neighbor field in the IIH contains its MAC address, and sets its neighbor status with Device A to **Up**.

### DIS Election

On a broadcast network, any two routers exchange information. If n routers are available on the network,  $n \times (n - 1)/2$  adjacencies must be established. Each status change of a router is transmitted to other routers, which wastes bandwidth resources. IS-IS resolves this problem by introducing the DIS. All routers send information to the DIS, which then broadcasts the network link status. Using the DIS and pseudo nodes simplifies network topology and reduces the length of LSPs generated by routers. When the network changes, fewer LSPs are generated. Therefore, fewer resources are consumed.

A DIS is elected after a neighbor relationship is established. Level-1 and Level-2 DISs are elected separately. You can configure different priorities for DISs at different levels. In DIS election, a Level-1 priority and a Level-2 priority are specified for every interface on every router. A router uses every interface to send IIHs and advertises its priorities in the IIHs to neighboring routers. The higher the priority, the higher the probability of being elected as the DIS. If there are multiple routers with the same highest priority on a broadcast network, the one with the largest MAC address is elected. The DISs at different levels can be the same router or different routers.

In the DIS election procedure, IS-IS is different from Open Shortest Path First (OSPF). In IS-IS, DIS election rules are as follows:

- The router with the priority of 0 also takes part in the DIS election.
- When a new router that meets the requirements of being a DIS is added to the broadcast network, the router is selected as the new DIS, which triggers a new round of LSP flooding.
- Establishment of a neighbor relationship on a P2P link

The establishment of a neighbor relationship on a P2P link is different from that on a broadcast link. A neighbor relationship on a P2P link can be established in 2-way or 3-way mode, as shown in [Table 1-81](#). By default, the 3-way handshake mechanism is used to establish a neighbor relationship on a P2P link.

**Table 1-81** Comparison between 2-way mode and 3-way mode

| Mode       | Description                                                                         | Advantages and Disadvantages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Reliability |
|------------|-------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 2-way mode | When a router receives an IIH, it unidirectionally sets up a neighbor relationship. | <p>Disadvantages:</p> <ul style="list-style-type: none"> <li>The unstable link status causes the loss of <b>complete sequence numbers protocol data units (CSNPs)</b> that are sent once an adjacency is set up. As a result, the link state databases (LSDBs) of two neighboring routers are not synchronized during the LSP update period.</li> <li>If two or more links exist between two routers, an adjacency can still be set up when one link is Down and another is Up in the same direction. A router that fails to detect the faulty link may also forward packets over this link.</li> </ul> | Low         |

| Mode       | Description                                                             | Advantages and Disadvantages                                                                                                                 | Reliability |
|------------|-------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| 3-way mode | A neighbor relationship is established after IIHs are sent three times. | Advantages: A neighbor relationship is established only when both ends are Up. This mechanism ensures that packets are transmitted securely. | High        |

### LSDB Synchronization

IS-IS is a link-state protocol. An IS-IS router obtains first-hand information from other routers running link-state protocols. Every router generates information about itself, directly connected networks, and links between itself and directly connected networks. The router then sends the generated information to other routers through adjacent routers. Every router saves link state information without modifying it. Finally, every router has the same network interworking information, and LSDB synchronization is complete. The process of synchronizing LSDBs is called LSP flooding. In LSP flooding, a router sends an LSP to its neighbors and the neighbors send the received LSP to their neighbors except the router that first sends the LSP. The LSP is flooded among the routers at the same level. This implementation allows each router at the same level to have the same LSP information and keep a synchronized LSDB.

All routers in the IS-IS routing domain can generate LSPs. A new LSP is generated in any of the following situations:

- Neighbor goes Up or Down.
- related interface goes Up or Down.
- Imported IP routes change.
- Inter-area IP routes change.
- A new metric value is configured for an interface.
- Periodic updates occur.

A router processes a received LSP as follows:

- Updating the LSDB on a broadcast link

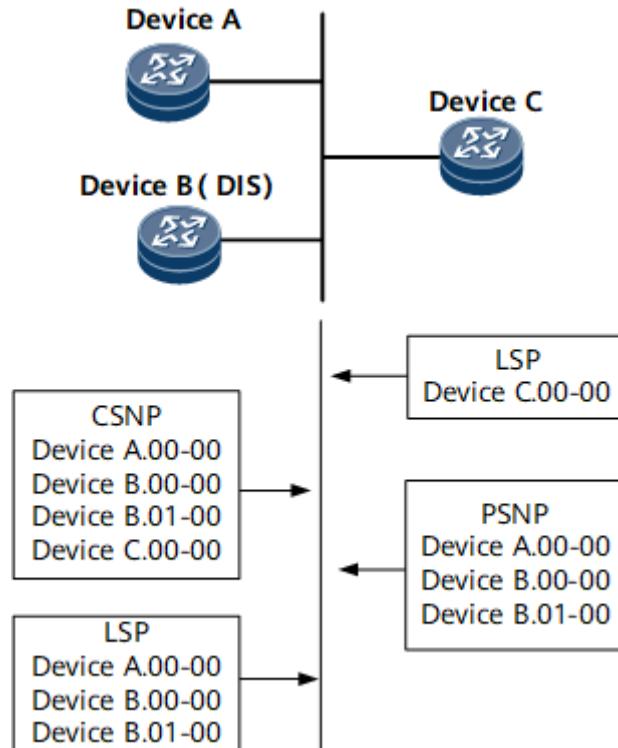
The DIS updates the LSDB to synchronize LSDBs on a broadcast network.

[Figure 1-228](#) shows the process of synchronizing LSDBs on a broadcast network.

- a. When the DIS receives an LSP, it searches the LSDB for the related records. If the DIS does not find the LSP in its LSDB, it adds the LSP to its LSDB and broadcasts the new LSDB.
- b. If the sequence number of the received LSP is greater than that of the local LSP, the DIS replaces the local LSP with the received LSP in the LSDB and broadcasts the new LSDB.

- c. If the sequence number of the received LSP is less than that of the local LSP, the DIS sends the local LSP in the LSDB to the inbound interface.
- d. If the sequence number of the received LSP is equal to that of the local LSP, the DIS compares the Remaining Lifetime of the two LSPs. If Remaining Lifetime of the received LSP is 0, the DIS replaces the LSP with the received LSP, and broadcasts the new LSDB. If the Remaining Lifetime of local LSP is 0, the DIS sends the LSP to the inbound interface.
- e. If the sequence number of the received LSP and the local LSP in the LSDB are the same and neither Remaining Lifetime is 0, the DIS compares the checksum of the two LSPs. If the received LSP has a greater checksum than that of the local LSP in the LSDB, the DIS replaces the local LSP in the LSDB with the received LSP and advertises the new LSDB. If the received LSP has a smaller checksum than that of the local LSP in the LSDB, the DIS sends the local LSP in the LSDB to the inbound interface.
- f. If the checksums of the received LSP and the local LSP are the same, the LSP is not forwarded.

**Figure 1-228** Process of updating the LSDB on a broadcast link



- Updating the LSDB on a P2P link
  - a. If the sequence number of the received LSP is greater than that of the local LSP in the LSDB, the router adds the received LSP to its LSDB. The router then sends a **PSNP packet** to acknowledge the received LSP and sends the LSP to all its neighbors except the neighbor that sends the LSP.
  - b. If the sequence number of the received LSP is less than that of the local LSP, the router directly sends its LSP to the neighbor and waits for a PSNP from the neighbor as an acknowledgement.

- c. If the sequence number of the received LSP is the same as that of the local LSP in the LSDB, the router compares the Remaining Lifetimes of the two LSPs. If Remaining Lifetime of the received LSP is 0, the router adds the LSP to its LSDB. The router then sends a PSNP to acknowledge the received LSP. If Remaining Lifetime of the local LSP is 0, the router directly sends the local LSP to the neighbor and waits for a PSNP from the neighbor.
- d. If the sequence number of the received LSP and the local LSP in the LSDB are the same, and neither Remaining Lifetime is 0, the router compares the checksum of the two LSPs. If the received LSP has a greater checksum than that of the local LSP, the router adds the received LSP to its LSDB. The router then sends a PSNP to acknowledge the received LSP. If the received LSP has a smaller checksum than that of the local LSP, the router directly sends the local LSP to the neighbor and waits for a PSNP from the neighbor. At last, the router sends the LSP to all its neighbors except the neighbor that sends the LSP.
- e. If the checksums of the received LSP and the local LSP are the same, the LSP is not forwarded.

### Route Calculation

When LSDB synchronization is complete and network convergence is implemented, IS-IS performs SPF calculation by using LSDB information to obtain the SPT. IS-IS uses the SPT to create a forwarding database (a routing table).

In IS-IS, link costs are used to calculate shortest paths. The default cost for an interface on a Huawei router is 10. The cost is configurable. The cost of a route is the sum of the cost of every outbound interface along the route. There may be multiple routes to a destination, among which the route with the smallest cost is the optimal route.

Level-1 routers can also calculate the shortest path to Level-2 routers to implement inter-area route selection. When a Level-1-2 router is connected to other areas, the router sets the value of the attachment (ATT) bit in its LSP to 1 and sends the LSP to neighboring routers. In the route calculation process, a Level-1 router selects the nearest Level-1-2 router as an intermediate router between the Level-1 and Level-2 areas.

### IS-IS Routing Information Control

IS-IS routes calculated using the SPF algorithm may bring about some problems. For example, too many routing entries slow down route lookup, or link usage is unbalanced. As a result, IS-IS routing cannot meet carriers' network planning and traffic management requirements.

To optimize IS-IS networks and facilitate traffic management, more precise route control is required. IS-IS uses the following methods to control routing information:

- [Route Leaking](#)
- [Route Summarization](#)
- [Load Balancing](#)
- [Administrative Tag](#)
- [IS-IS Mesh Group](#)

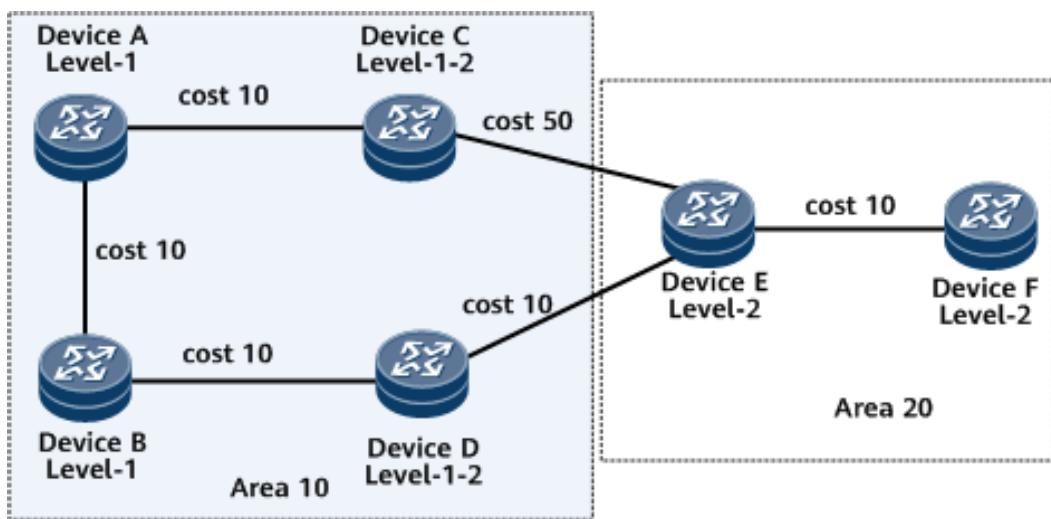
- [Link-group](#)

### Route Leaking

When Level-1 and Level-2 areas both exist on an IS-IS network, Level-2 routers do not advertise the learned routing information about a Level-1 area and the backbone area to any other Level-1 area by default. Therefore, Level-1 routers do not know the routing information beyond the local area. As a result, the Level-1 routers cannot select the optimal routes to the destination beyond the local area.

With route leaking, Level-1-2 routers can select routes using routing policies, or tags and advertise the selected routes of other Level-1 areas and the backbone area to the Level-1 area. [Figure 1-229](#) shows the typical networking for route leaking.

**Figure 1-229** Typical networking for route leaking



- Device A, Device B, Device C, and Device D belong to area 10. Device A and Device B are Level-1 routers. Device C and Device D are Level-1-2 routers.
- Device E and Device F belong to area 20 and are Level-2 routers.

If Device A sends a packet to Device F, the selected optimal route should be Device A -> Device B -> Device D -> Device E -> Device F because its cost is 40 ( $10 + 10 + 10 + 10 = 40$ ) which is less than that of Device A -> Device C -> Device E -> Device F ( $10 + 50 + 10 = 70$ ). However, if you check routes on Device A, you can find that the selected route is Device A -> Device C -> Device E -> Device F, which is not the optimal route from Device A to Device F.

This is because Device A does not know the routes beyond the local area, and therefore, the packets sent by Device A to other network segments are sent through the default route generated by the nearest Level-1-2 device.

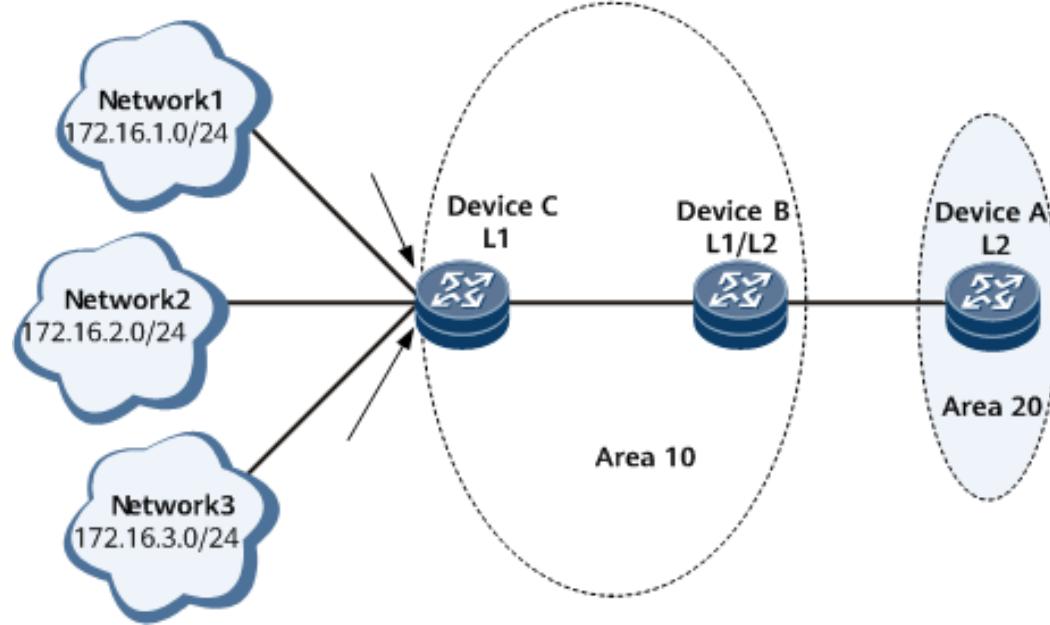
In this case, you can enable route leaking on the Level-1-2 devices (Device C and Device D). Then, check the route and you can find that the selected route is Device A -> Device B -> Device D -> Device E -> Device F.

### Route Summarization

On a large-scale IS-IS network, links connected to devices within an IP address range may alternate between up and down. With route summarization, multiple

routes with the same IP prefix are summarized into one route, which prevents route flapping, reduces routing entries and system resource consumption, and facilitates route management. [Figure 1-230](#) shows the typical networking.

[Figure 1-230](#) Typical networking for route summarization



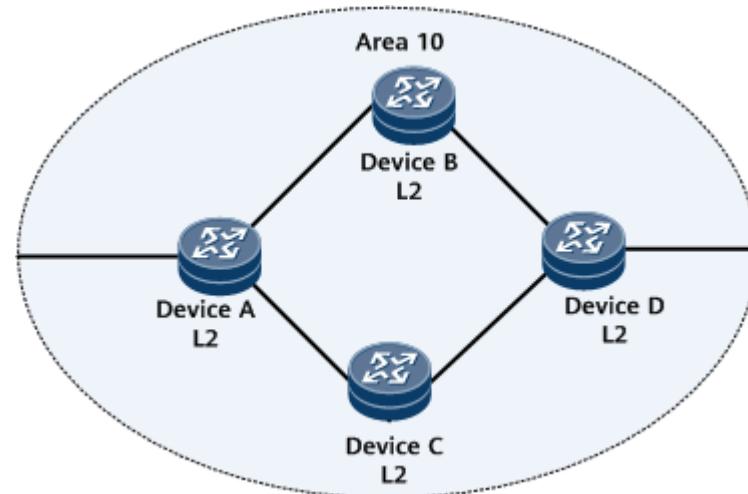
- router A, router B, and router C use IS-IS to communicate with each other.
- Device A belongs to area 20, and Device B and Device C belong to area 10.
- Device A is a Level-2 router. Device B is a Level-1-2 router. Device C is a Level-1 router.
- Device B maintains Level-1 and Level-2 LSDBs and leaks the routes to three network segments (172.16.1.0/24, 172.16.2.0/24, and 172.16.3.0/24) from the Level-1 area to the Level-2 area. If a link fault causes the Device C interface with IP address 172.16.1.1/24 to frequently alternate between up and down, the state change is advertised to the Level-2 area, triggering frequent LSP flooding and SPF calculation on Device A. As a result, the CPU usage on Device A increases, and even network flapping occurs.

If Device B is configured to summarize routes to the three network segments in the Level-1 area into route 172.16.0.0/22, the number of routing entries on Device B is reduced; in addition, the impact of link state changes in the Level-1 area on route convergence in the Level-2 area can be reduced.

## Load Balancing

When multiple equal-cost routes are available on a network, you can configure IS-IS load balancing to improve link utilization and prevent network congestion caused by link overload. IS-IS load balancing evenly distributes traffic among multiple equal-cost paths. [Figure 1-231](#) shows the typical networking for load balancing.

Figure 1-231 Typical networking for load balancing



- Device A, Device B, Device C, and Device D communicate with each other on an IP network using IS-IS.
- Device A, Device B, Device C, and Device D belong to area 10 and are Level-2 routers.
- If load balancing is not enabled, traffic on Device A is transmitted along the optimal route obtained using the SPF calculation. Consequently, traffic on different links is unbalanced. Enabling load balancing on Device A sends traffic to routerDevice D through routerDevice B and Device C. This transmission mode relieves the load on the optimal route.

Load balancing supports per-packet load balancing and per-flow load balancing. For details, see *NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X Feature Description - IP Routing*.

IS-IS supports not only intra-process load balancing, but also inter-process load balancing when equal-cost routes exist between different processes.

## Administrative Tag

Administrative tags carry administrative information about IP address prefixes. When the cost type is wide, wide-compatible, or compatible and the prefix of the reachable IP address to be advertised by IS-IS has this cost type, IS-IS adds the administrative tag to the reachability type-length-value (TLV) in the prefix. In this manner, the administrative tag is advertised to the entire routing domain along with the prefix so that routes can be imported or filtered based on the administrative tag.

## IS-IS Mesh Group

As defined in IS-IS, upon receipt of a new LSP, a router floods it. On a network with high connectivity and multiple P2P links, this causes repeated LSP flooding and wastes bandwidth resources. To prevent this problem, you can configure a mesh group to reduce bandwidth waste.

A mesh group consists of a group of interfaces. Interfaces in a mesh group flood the LSPs received from the local group only to the interfaces in other mesh groups

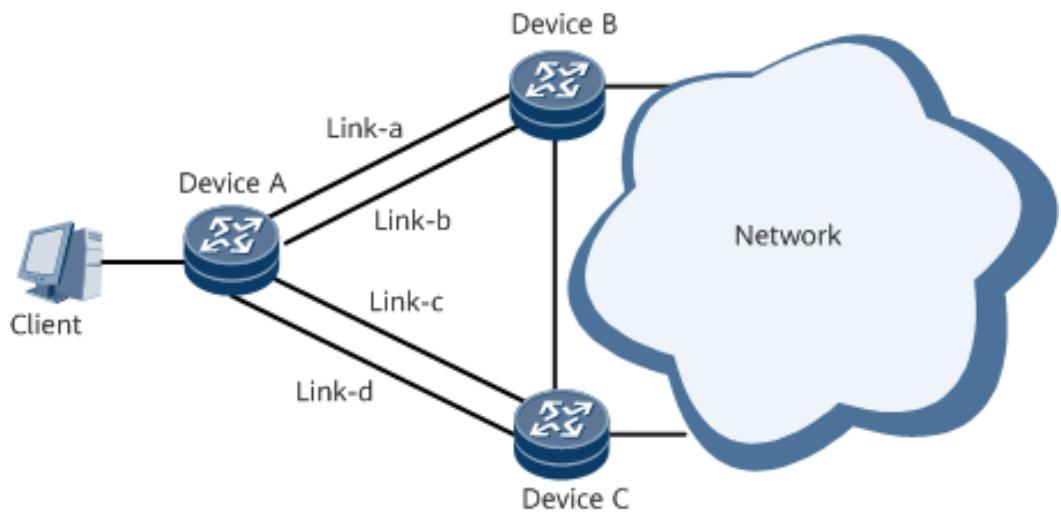
and those that are not in any mesh group. In addition, interfaces in a mesh group use the CSNP and PSNP mechanisms to implement LSDB synchronization in the entire network segment.

## link-group

In [Figure 1-232](#), router A is dual-homed to the IS-IS network through router B and router C. The path router A -> router B is primary and the path router A -> router C is backup. The bandwidth of each link is 100 Gbit/s, and the traffic from Client is transmitted at 150 Gbit/s. In this situation, both links in the path router A -> router B or the path router A -> router C need to carry the traffic. If Link-a fails, Link-b takes over all the traffic. However, the bandwidth of Link-b is not sufficient to carry the traffic. As a result, traffic loss occurs.

To address this problem, configure link groups. You can add multiple links to a link group. If one of the links fails and the bandwidth of the other links in the group is not sufficient to carry the traffic, the link group automatically increases the costs of the other links to a configured value so that this link group is not selected. Then, traffic is switched to another link group.

**Figure 1-232** IS-IS dual-homing access networking



In [Figure 1-232](#), Link-a and Link-b belong to link group 1, and Link-c and Link-d belong to link group 2.

- If Link-a fails, link group 1 automatically increases the cost of Link-b so that the traffic is switched to Link-c and Link-d.
- If both Link-a and Link-c fail, the link groups increase the costs of Link-b and Link-d (to the same value) so that Link-b and Link-d load-balance the traffic.

## IS-IS Neighbor Relationship Flapping Suppression

IS-IS neighbor relationship flapping suppression works by delaying neighbor relationship reestablishment or setting the link cost to the maximum value.

## Background

If the status of an interface carrying IS-IS services alternates between Up and Down, IS-IS neighbor relationship flapping occurs on the interface. During the

flapping, IS-IS frequently sends Hello packets to reestablish the neighbor relationship, synchronizes LSDBs, and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, IS-IS services, and other IS-IS-dependent services, such as LDP and BGP. IS-IS neighbor relationship flapping suppression can address this problem by delaying IS-IS neighbor relationship reestablishment or preventing service traffic from passing through flapping links.

## Related Concepts

**Flapping\_event:** reported when the status of a neighbor relationship on an interface last changes from **Up** to **Init** or **Down**. The flapping\_event triggers flapping detection.

**Flapping\_count:** number of times flapping has occurred.

**Detect-interval:** interval at which flapping is detected. The interval is used to determine whether to trigger a valid flapping\_event.

**Threshold:** flapping suppression threshold. When the flapping\_count exceeds the threshold, flapping suppression takes effect.

**Resume-interval:** interval used to determine whether flapping suppression exits. If the interval between two valid flapping\_events is longer than the resume-interval, flapping suppression exits.

## Implementation

### Flapping detection

IS-IS interfaces start a flapping counter. If the interval between two consecutive flapping\_events is shorter than the detect-interval, a valid flapping\_event is recorded, and the flapping\_count increases by 1. When the flapping\_count reaches or exceeds the threshold, the system determines that flapping occurs, implements flapping suppression, and sets the flapping\_count to 0. If the interval between two valid flapping\_events is longer than the resume-interval before the flapping\_count reaches the threshold again, the system sets the flapping\_count to 0 again. Interfaces start the suppression timer when the status of a neighbor relationship last changes to **Init** or **Down**.

The detect-interval, threshold, and resume-interval are configurable.

### Flapping suppression

Flapping suppression works in either Hold-down or Hold-max-cost mode.

- Hold-down mode: In the case of frequent flooding and topology changes during neighbor relationship establishment, interfaces prevent neighbor relationships from being reestablished during the suppression period, which minimizes LSDB synchronization attempts and packet exchanges.
- Hold-max-cost mode: If the traffic forwarding path frequently changes, interfaces use the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) of the flapping link during the suppression period to prevent traffic from passing through the flapping link.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

By default, IS-IS uses the Hold-max-cost mode. The mode and suppression period can be changed manually.

 **NOTE**

When an interface enters the flapping suppression state, all neighbor relationships on the interface enter the state accordingly.

### Exiting from flapping suppression

Interfaces exit from flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding IS-IS process is reset.
- A command is run to exit from flapping suppression.
- Three Hello packets in which the padding TLV carries a sub-TLV with the value being 251 are sent consecutively to instruct the peer device to exit flapping suppression.

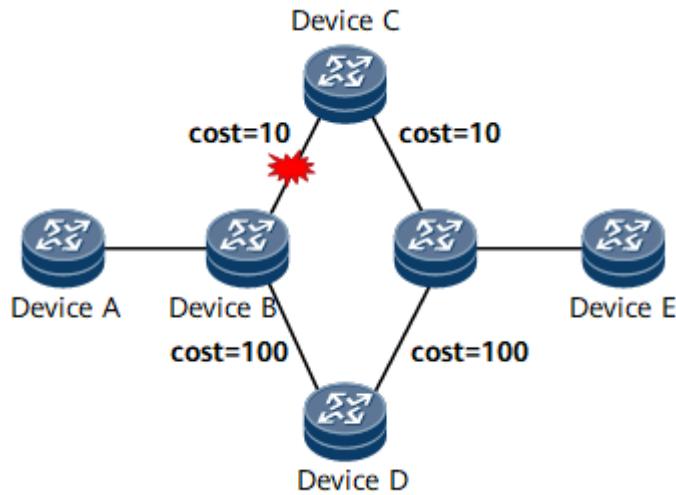
## Typical Usage Scenario

### Basic scenario

On the network shown in [Figure 1-233](#), the traffic forwarding path is Device A -> Device B -> Device C -> Device E before a link failure occurs. After the link between Device B and Device C fails, the forwarding path switches to Device A -> Device B -> Device D -> Device E. If the neighbor relationship between Device B and Device C frequently flaps at the early stage of the path switchover, the forwarding path for the traffic from Device A to Device E will be switched frequently, causing traffic loss and affecting network stability. If the neighbor flapping between Device B and Device C meets flapping suppression conditions, flapping suppression is triggered.

- If flapping suppression works in Hold-down mode, the neighbor relationship between Device B and Device C is prevented from being reestablished during the suppression period, in which traffic is forwarded along the path Device A -> Device B -> Device D -> Device E.
- If flapping suppression works in Hold-max-cost mode, the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) is used as the cost of the link between Device B and Device C during the suppression period, and traffic is forwarded along the path Device A -> Device B -> Device D -> Device E.

Figure 1-233 Flapping suppression in a basic scenario



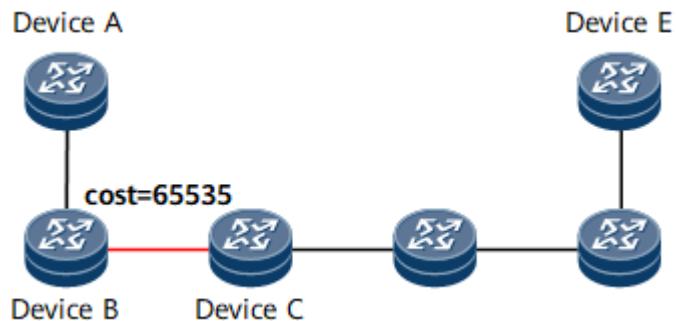
#### Single-forwarding path scenario

When only one forwarding path exists on the network, the flapping of the neighbor relationship between any two devices on the path will interrupt traffic forwarding. In [Figure 1-234](#), the only traffic forwarding path is Device A -> Device B -> Device C -> Device E. If the neighbor relationship between Device B and Device C flaps, and the flapping meets suppression conditions, flapping suppression takes effect. However, the link between Device B and Device C is a key link on the network. Once the link is disconnected, the entire network is divided into two networks that are not interconnected. Therefore, Hold-max-cost mode (rather than Hold-down mode) is recommended in this scenario. If flapping suppression works in Hold-max-cost mode, the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) is used as the cost of the link between Device B and Device C during the suppression period. After the network becomes stable and the suppression timer expires, flapping suppression exits automatically, and the service link recovers immediately.

#### NOTE

By default, the Hold-max-cost mode takes effect.

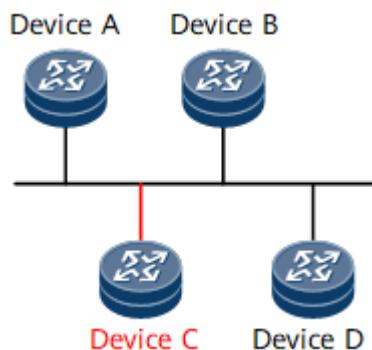
Figure 1-234 Flapping suppression in a single-forwarding path scenario



#### Broadcast scenario

In [Figure 1-235](#), four devices are deployed on the same broadcast network using switches, and the devices are broadcast network neighbors. If Device C keeps flapping due to a link failure, and Device A and Device B were deployed at different time (Device A was deployed earlier for example) or the flapping suppression parameters on Device A and Device B are different, Device A first detects the flapping and suppresses Device C. Consequently, the Hello packets sent by Device A do not carry Device C's router ID. However, Device B has not detected the flapping yet and still considers Device C a valid node. As a result, the DIS candidates identified by Device A are Device B and Device D, whereas the DIS candidates identified by Device B are Device A, Device C, and Device D. Different candidates result in different election results, which may lead to route calculation errors. In scenarios where an interface has multiple neighbors, such as on a broadcast, P2MP, or NBMA network, all neighbors on the interface are suppressed if the last flapping event that the neighbor status on the interface becomes Init or Down is detected (flapping detection cannot be performed by neighbor). Specifically, if Device C flaps, Device A, Device B, and Device D on the broadcast network are all suppressed. After the network stabilizes and the suppression timer expires, Device A, Device B, and Device D are restored to normal status.

**Figure 1-235** Flapping suppression on a broadcast network



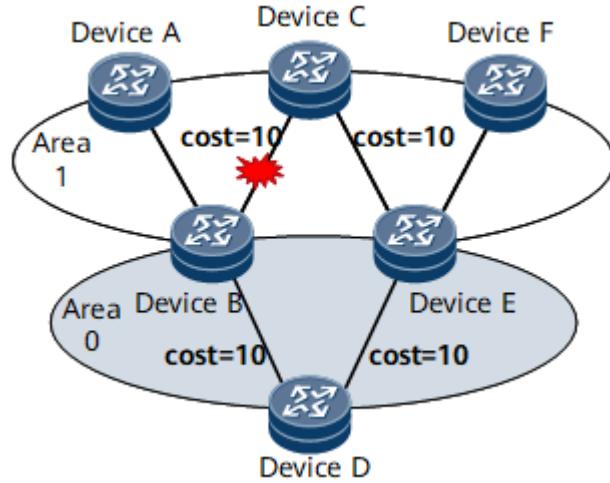
### Scenario of multi-level networking

In [Figure 1-236](#), Device A, Device B, Device C, Device E, and Device F are Level-1 neighbors in area 1. Device B, Device D, and Device E are Level-2 neighbors in area 0. Traffic from Device A to Device F is preferentially forwarded along an intra-area route, and the forwarding path is Device A -> Device B -> Device C -> Device E -> Device F. If the neighbor relationship between Device B and Device C flaps and the flapping meets suppression conditions, flapping suppression takes effect in the default mode (Hold-max-cost). However, the forwarding path remains unchanged (Device A -> Device B -> Device C -> Device E -> Device F) after the neighbor flapping occurs because intra-area routes take precedence over inter-area routes during route selection regardless of costs according to IS-IS route selection rules. The Hold-max-cost mode cannot suppress traffic path switching in this case. To prevent traffic loss in multi-area scenarios, configure Hold-down mode to prevent the neighbor relationship between Device B and Device C from being reestablished during the suppression period. During this period, traffic is forwarded along the path Device A -> Device B -> Device D -> Device E -> Device F.

 NOTE

By default, the Hold-max-cost mode is enabled for IS-IS. You can change the mode to Hold-down using a command.

**Figure 1-236 Flapping suppression in a multi-level scenario**



**Scenario with both LDP-IGP synchronization and neighbor relationship flapping suppression configured**

In [Figure 1-237](#), if the link between PE1 and P1 fails, an LDP LSP switchover is implemented immediately, causing the original LDP LSP to be deleted before a new LDP LSP is established. To prevent traffic loss, IGP-LDP synchronization needs to be configured. With IGP-LDP synchronization, the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) is used as the cost of the new LSP to be established. After the new LSP is established, the original cost takes effect. Consequently, the original LSP is deleted, and LDP traffic is forwarded along the new LSP.

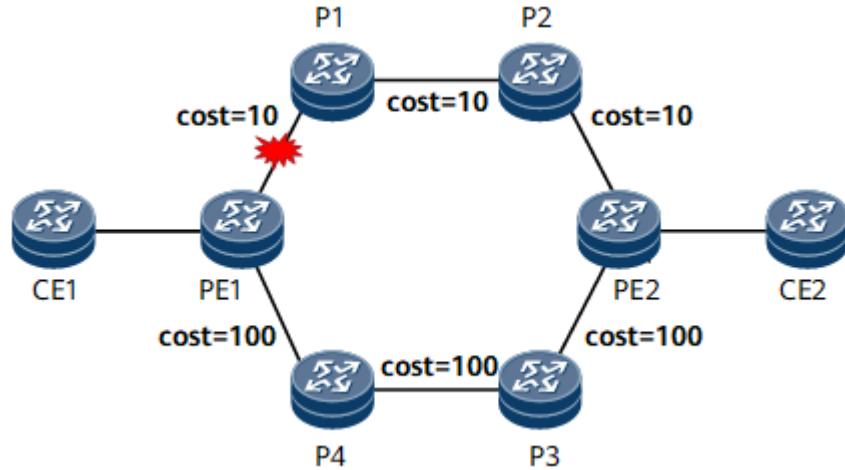
LDP-IGP synchronization and OSPF neighbor relationship flapping suppression work in either Hold-down or Hold-max-cost mode. If both functions are configured, Hold-down mode takes precedence over Hold-max-cost mode, followed by the configured link cost. [Table 1-82](#) lists the suppression modes that take effect in different situations.

**Table 1-82** Principles for selecting the suppression modes that take effect in different situations

| LDP-IGP Synchronization/IS-IS Neighbor Relationship Flapping Suppression Mode | LDP-IGP Synchronization Hold-down Mode | LDP-IGP Synchronization Hold-max-cost Mode | Exited from LDP-IGP Synchronization Suppression                                    |
|-------------------------------------------------------------------------------|----------------------------------------|--------------------------------------------|------------------------------------------------------------------------------------|
| IS-IS Neighbor Relationship Flapping Suppression Hold-down Mode               | Hold-down                              | Hold-down                                  | Hold-down                                                                          |
| IS-IS Neighbor Relationship Flapping Suppression Hold-max-cost Mode           | Hold-down                              | Hold-max-cost                              | Hold-max-cost                                                                      |
| Exited from IS-IS Neighbor Relationship Flapping Suppression                  | Hold-down                              | Hold-max-cost                              | Exited from LDP-IGP synchronization and neighbor relationship flapping suppression |

For example, the link between PE1 and P1 frequently flaps in [Figure 1-237](#), and both LDP-IGP synchronization and IS-IS neighbor relationship flapping suppression are configured. In this case, the suppression mode is selected based on the preceding principles. No matter which mode (Hold-down or Hold-max-cost) is selected, the forwarding path is PE1 -> P4 -> P3 -> PE2.

**Figure 1-237 Scenario with both LDP-IGP synchronization and neighbor relationship flapping suppression configured**



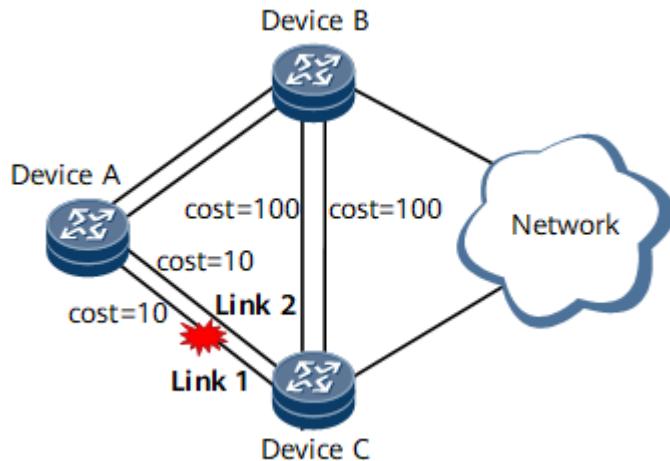
**Scenario with both bit-error-triggered protection switching and IS-IS neighbor relationship flapping suppression configured**

If a link has poor link quality, services transmitted along it may be adversely affected. If bit-error-triggered protection switching is configured and the bit error rate (BER) along a link exceeds a specified value, a bit error event is reported, and the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) is used as the cost of the link, triggering route reselection. Consequently, service traffic is switched to the backup link. If both bit-error-triggered protection switching and IS-IS neighbor relationship flapping suppression are configured, they both take effect. Hold-down mode takes precedence over Hold-max-cost mode, followed by the configured link cost.

**Scenario with both Link-bundle and IS-IS neighbor relationship flapping suppression configured**

When the service traffic rate exceeds the capacity of the link, multiple links must be used. In such cases, if one link fails, traffic is switched to another link and excess traffic is discarded due to the new link's limited forwarding capacity. Link-bundle can address this problem. If the number of faulty links reaches the upper threshold, the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) is used as the cost of all links in the link bundle to switch all service traffic to the backup nodes. If both link-bundle and neighbor relationship flapping suppression are configured and the number of flapping links reaches the upper threshold, the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) must be configured as the cost of all other links in the link bundle to prevent service loss caused by user traffic congestion. In [Figure 1-238](#), two parallel links forward traffic between Device A and Device C. If link 1 fails, link 2 is incapable of bearing all service traffic. If both link-bundle and neighbor relationship flapping suppression are configured and Link 1 flaps, the maximum cost (16777214 in IS-IS wide mode and 63 in IS-IS narrow mode) must be configured for Link 2 to avoid service traffic congestion. Only the Hold-max-cost mode therefore can be configured for neighbor relationship flapping suppression to switch the traffic forwarding path to Device A->Device B->Device C.

**Figure 1-238** Scenario with both Link-bundle and IS-IS neighbor relationship flapping suppression configured



## IS-IS Overload

The overload (OL) field of LSPs configured on a device prevents other devices from calculating the routes passing through this device.

If a system fails to store new LSPs for LSDB synchronization, the routes calculated by the system are incorrect. In that case, the system enters the Overload state. The user can configure the device to enter the Overload state when the system lacks sufficient memory. At present, users can set the Overload timer when IS-IS is started and configure whether to delete the leaked routes and whether to advertise the imported routes. A device enters the Overload state after an exception occurs on the device or when it is configured to enter the state.

- If IS-IS enters the Overload state after an exception occurs on the device, the system deletes all imported or leaked routes.
  - If IS-IS enters the Overload state based on a user configuration, the system only deletes all imported or leaked routes if configured to do so.

Although LSPs with overload fields are flooded throughout the network, they are ignored in the calculation of the routes passing through the device in the Overload state. Specifically, after the overload field of LSPs is configured on a device, other devices do not count the routes that pass through the device when performing SPF calculation, but the direct routes between the device and other devices are still calculated.

If a device in an IS-IS domain is faulty, routes may be incorrectly calculated across the entire domain. The overload field can be configured for the device to isolate it from the IS-IS network temporarily, which facilitates fault isolation.

## IS-IS Fast Convergence

IS-IS fast convergence is an extended feature of IS-IS implemented to speed up route convergence.

- Incremental SPF (I-SPF)

I-SPF recalculates only the routes of the changed nodes rather than the routes of all nodes when the network topology changes, which speeds up the calculation of routes.

- Partial Route Calculation (PRC)

PRC calculates only those routes which have changed when the network topology changes.

- Link State PDUs (LSP) fast flooding

LSP fast flooding speeds up LSP flooding.

- Intelligent timer

The first timeout period of the timer is fixed. If an event that triggers the timer occurs before the set timer expires, the next timeout period of the timer increases.

The intelligent timer applies to LSP generation and SPF calculation.

## I-SPF

In ISO 10589, the Dijkstra algorithm was adopted to calculate routes. When a node changes on the network, the algorithm recalculates all routes. The calculation requires a long time to complete and consumes a significant amount of CPU resources, reducing convergence speed.

I-SPF improves the algorithm. Except for the first time the algorithm is run, only the nodes that have changed rather than all nodes in the network are used in the calculation. The SPT generated using I-SPF is the same as that generated using the previous algorithm. This significantly decreases CPU usage and speeds up network convergence.

## PRC

Similar to I-SPF, PRC calculates only routes that have changed. PRC, however, does not calculate the shortest path. It updates routes based on the SPT calculated by I-SPF.

In route calculation, a leaf represents a route, and a node represents a device. Either an SPT change or a leaf change causes a routing information change. The SPT change is irrelevant to the leaf change. PRC processes routing information as follows:

- If the SPT changes after I-SPF calculation, PRC calculates all the leaves only on the changed node.
- If the SPT remains unchanged after I-SPF calculation, PRC calculates only the changed leaves.

For example, if a new route is imported, the SPT of the entire network remains unchanged. In this case, PRC updates only the interface route for this node, thereby reducing the CPU usage.

PRC working with I-SPF further improves network convergence performance and replaces the original SPF algorithm.

 NOTE

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, only I-SPF and PRC are used to calculate IS-IS routes.

## LSP Fast Flooding

When an IS-IS device receives new LSPs from other devices, it updates the LSPs in the LSDB and periodically floods the updated LSPs based on a timer. Therefore, the synchronization of all LSDBs is slow.

LSP fast flooding can address this problem. With this function configured, when the router receives LSPs that can trigger route calculation or route update, the router makes its best efforts to flood these LSPs before route calculation is performed. This speeds up LSDB synchronization on the entire network. This flooding mode significantly speeds up the network-wide convergence speed.

 NOTE

LSP fast flooding is supported by default and does not need to be configured.

## Intelligent Timer

Although the route calculation algorithm is improved, the long interval for triggering route calculation also affects the convergence speed. A millisecond-level timer can shorten the interval. Frequent network changes, however, also consume too much CPU resources. The SPF intelligent timer can quickly respond to a few external emergencies and avoid excessive CPU usage.

In most cases, an IS-IS network running normally is stable. The frequent changes on a network are rather rare, and IS-IS does not calculate routes frequently. Therefore, a short period (within milliseconds) can be configured as the first interval for route calculation. If the network topology changes frequently, the interval set by the intelligent timer increases with the calculation times to reduce CPU consumption.

The LSP generation intelligent timer is similar to the SPF intelligent timer. When the LSP generation intelligent timer expires, the system generates a new LSP based on the current topology. In the original implementation mechanism, a timer with a fixed interval is used, which cannot meet the requirements of fast convergence and low CPU usage at the same time. Therefore, the LSP generation timer is designed as an intelligent timer to respond to emergencies (for example, the interface goes Up or Down) quickly and speed up network convergence. In addition, when the network changes frequently, the interval for the intelligent timer becomes longer to reduce CPU consumption.

## IS-IS LSP Fragment Extension

If the LSP capacity is insufficient, newly imported routes and new TLVs fail to be added to LSP fragments. In this case, you can use LSP fragment extension to increase the LSP capacity, restoring the LSP space. When the LSP space is restored, the system automatically attempts to re-add these routes and TLVs to LSP fragments.

When the LSPs to be advertised by IS-IS contain a large amount of information, they are advertised in multiple Link State PDUs (LSP) fragments belonging to the same system.

Virtual system IDs can be configured, and virtual LSPs that carry routing information can be generated for IS-IS.

IS-IS LSP fragment extension allows an IS-IS device to generate more LSP fragments and carry more IS-IS information.

## Terms

- Originating system

The originating system is a device that runs the IS-IS protocol. A single IS-IS process advertises LSPs as virtual devices do, except that the originating system refers to a real IS-IS process.

- Normal system ID

The normal system ID is the system ID of the originating system.

- Additional system ID

The additional system ID, assigned by the network administrator, is used to generate additional or extended LSP fragments. A maximum of 256 additional or extended LSP fragments can be generated. Like a normal system ID, an additional system ID must be unique in a routing domain.

- Virtual system

The virtual system, identified by an additional system ID, is used to generate extended LSP fragments. These fragments carry additional system IDs in their LSP IDs.

## Principles

IS-IS LSP fragments are identified by the LSP Number field in their LSP IDs. The LSP Number field is 1 byte. Therefore, an IS-IS process can generate a maximum of 256 fragments. With fragment extension, more information can be carried.

Each system ID represents a virtual system, and each virtual system can generate 256 LSP fragments. In addition, another virtual systems can be configured. Therefore, an IS-IS process can generate more LSP fragments.

After a virtual system and fragment extension are configured, an IS-IS device adds the contents that cannot be contained in its LSPs to the LSPs of the virtual system and notifies other devices of the relationship between the virtual system and itself through a special TLV in the LSPs.

## IS Alias ID TLV

Standard protocol defines a special Type-Length-Value (TLV): IS Alias ID TLV.

**Table 1-83 IS Alias ID TLV**

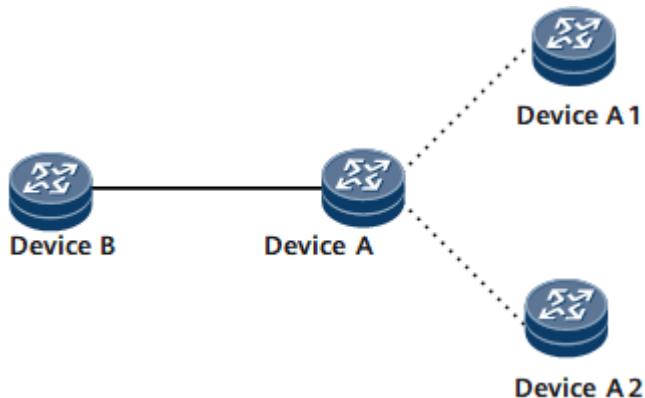
| Field             | Length         | Description                                                     |
|-------------------|----------------|-----------------------------------------------------------------|
| Type              | 1 byte         | TLV type. If the value is 24, it indicates the IS Alias ID TLV. |
| Length            | 1 byte         | TLV length.                                                     |
| System ID         | 6 bytes        | System ID.                                                      |
| Pseudonode number | 1 byte         | Pseudonode number.                                              |
| Sub-TLVs length   | 1 byte         | Length of sub-TLVs.                                             |
| Sub-TLVs          | 0 to 247 bytes | Sub-TLVs.                                                       |

LSPs with fragment number 0 sent by the originating system and virtual system carry IS Alias ID TLVs to indicate the originating system.

## Operation Modes

IS-IS devices can use the LSP fragment extension feature in the following modes:

**Figure 1-239 Networking for IS-IS LSP fragment extension**



- Mode-1

Mode-1 is used when some devices on the network do not support LSP fragment extension.

In this mode, virtual systems participate in SPF calculation. The originating system advertises LSPs containing information about links to each virtual system and each virtual system advertises LSPs containing information about links to the originating system. In this manner, the virtual systems function the same as the actual devices connected to the originating system on the network.

Mode-1 is a transitional mode for earlier versions that do not support LSP fragment extension. In the earlier versions, IS-IS cannot identify Alias ID TLVs. Therefore, the LSP sent by a virtual system must look like a common IS-IS LSP.

The LSP sent by a virtual system contains the same area address and overload bit as those in the common LSP. If the LSPs sent by a virtual system contain TLVs specified in other features, the TLVs must be the same as those in common LSPs.

LSPs sent by a virtual system carry information of the neighbor (the originating system), and the carried cost is the maximum value minus 1. LSPs sent by the originating system carry information of the neighbor (the virtual system), and the carried cost is 0. This mechanism ensures that the virtual system is a node downstream of the originating system when other devices calculate routes.

In [Figure 1-239](#), Device B does not support LSP fragment extension; Device A supports LSP fragment extension in mode-1; Device A1 and Device A2 are virtual systems of Device A. Device A1 and Device A2 send LSPs carrying partial routing information of Device A. After receiving LSPs from Device A, Device A1, and Device A2, Device B considers there to be three devices at the peer end and calculates routes normally. Because the cost of the route from Device A to Device A1 or Device A2 is 0, the cost of the route from Device B to Device A is equal to that from Device B to Device A1.

- Mode-2

Mode-2 is used when all the devices on the network support LSP fragment extension. In this mode, virtual systems do not participate in SPF calculation. All the devices on the network know that the LSPs generated by the virtual systems actually belong to the originating system.

IS-IS working in mode-2 identifies IS Alias ID TLVs, which are used to calculate the SPT and routes.

In [Figure 1-239](#), Device B supports LSP fragment extension, and Device A supports LSP fragment extension in mode-2; Device A1 and Device A2 send LSPs carrying some routing information of Device A. After receiving LSPs from Device A1 and Device A2, Device B obtains IS Alias ID TLV and learns that the originating system of Device A1 and Device A2 is Device A. Device B then considers information advertised by Device A1 and Device A2 to be about Device A.

Whatever the LSP fragment extension mode, LSPs can be resolved. However, if LSP fragment extension is not supported, only LSPs in mode-1 can be resolved.

**Table 1-84** Comparison between mode-1 and mode-2

| LSP Field              | Carried in Mode-1 | Carried in Mode-2 |
|------------------------|-------------------|-------------------|
| IS Alias ID            | Yes               | Yes               |
| Area                   | Yes               | No                |
| Overload bit           | Yes               | Yes               |
| IS NBR/IS EXTENDED NBR | Yes               | No                |

| LSP Field | Carried in Mode-1 | Carried in Mode-2 |
|-----------|-------------------|-------------------|
| Routing   | Yes               | Yes               |
| ATT bit   | Yes, with value 0 | Yes, with value 0 |
| P bit     | Yes, with value 0 | Yes, with value 0 |

## Process

After LSP fragment extension is configured, if information is lost because LSPs overflow, the system restarts the IS-IS process. After being restarted, the originating system loads as much routing information as possible. Any excessive information beyond the forwarding capability of the system is added to the LSPs of the virtual systems for transmission. In addition, if a virtual system with routing information is deleted, the system automatically restarts the IS-IS process.

## Usage Scenario

### NOTE

If there are non-Huawei devices on the network, LSP fragment extension must be set to mode-1. Otherwise, these devices cannot identify LSPs.

Configuring LSP fragment extension and virtual systems before setting up IS-IS neighbors or importing routes is recommended. If IS-IS neighbors are set up or routes are imported first and the information to be carried exceeds the forwarding capability of 256 fragments before LSP fragment extension and virtual systems are configured, you have to restart the IS-IS process for the configurations to take effect.

## IS-IS 3-Way Handshake

IS-IS introduces the 3-way handshake mechanism on P2P links to ensure a reliable data link layer.

Based on ISO 10589, the IS-IS 2-way handshake mechanism uses Hello packets to set up P2P adjacencies between neighboring devices. When a device receives a Hello packet from the other end, it regards the other end as Up and sets up an adjacency with it. However, this mechanism has some serious shortcomings.

When two or more links exist between two devices, an adjacency can still be set up where one link is Down and the other is Up in the same direction. The parameters of the other link are used in SPF calculation. As a result, a device that does not detect any fault along the faulty link will continue trying to forward packets over the link.

The 3-way handshake mechanism resolves these problems on P2P links. In 3-way handshake mode, a device regards a neighbor Up and sets up an adjacency with it only after confirming that the neighbor has received the packet that the device sends.

In addition, the 3-way handshake mechanism uses the 32-bit Extended Local Circuit ID field, which extends the original 8-bit Extended Local Circuit ID field and the limit of only 255 P2P links.

 NOTE

By default, the IS-IS 3-way handshake mechanism is implemented on P2P links.

## IS-IS Extended Prefix Attribute

IS-IS defines a type of extended prefix attribute (IPv4/IPv6 Extended Reachability Attribute) sub-TLVs. These sub-TLVs are used to describe the origin of advertised routes and can be advertised in IPv4, IPv6, and locator TLVs. Based on the advertised sub-TLVs, the device can determine whether the received routes are inter-domain routes or host routes.

The extended prefix attribute flag (IPv4/IPv6 Extended Reachability Attribute Flags) is a part of the extended prefix attribute sub-TLV. The specific composition is as follows:

**Figure 1-240** Format of the IPv4/IPv6 Extended Reachability Attribute Flags sub-TLV



**Table 1-85** describes the fields of the extended prefix attribute flag.

**Table 1-85** Description of fields in the IPv4/IPv6 Extended Reachability Attribute Flags sub-TLV

| Field | Length | Meaning                                                                                                                                                                                                                                                                                                                                                                          |
|-------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| X     | 1 bit  | External prefix flag: <ul style="list-style-type: none"><li>This flag indicates whether the prefix is imported from another protocol or IS-IS process and re-advertised.</li><li>The value of this flag remains unchanged during inter-level leaking.</li><li>For an IPv6 prefix, the X flag is not set to 1 because the parent TLV has a flag with the same function.</li></ul> |
| R     | 1 bit  | Re-advertisement flag: <ul style="list-style-type: none"><li>This flag is set to 1 when routes leak from a Level-1 area to a Level-2 area or from a Level-2 area to a Level-1 area on an IS-IS Level-1-2 device.</li><li>When local direct routes leak, this flag is not set to 1.</li></ul>                                                                                     |

| Field | Length | Meaning                                                                                                                                                                                                                                                                                                                                                                       |
|-------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| N     | 1 bit  | <p>Node flag:</p> <ul style="list-style-type: none"><li>• This flag can be set to 1 when an IPv4 host route (with a 32-bit subnet mask) or IPv6 host route (with a 128-bit subnet mask) is advertised. You can run a command to determine whether to set this flag.</li><li>• The setting of this flag is kept when routes are leaked on an IS-IS Level-1-2 device.</li></ul> |

## IS-IS NSR

### Background

As networks develop, the demand for data, audio, and video services is growing, which imposes increasing requirements on IP network reliability. If an AMB/SMB switchover is performed on a device due to a maintenance operation or a single point of failure, routes may fail to converge, which may result in traffic loss or even a network breakdown. Non-stop routing (NSR) can address this problem and ensure uninterrupted forwarding of key services.

### Related Concepts

- High availability (HA): supports data backup between the AMB and SMB.
- Non-stop forwarding (NSF): enables a node to use the GR mechanism to ensure uninterrupted transmission during an AMB/SMB switchover.
- NSR: allows a standby control plane to take over traffic from an active control plane if the active control plane fails, preventing the control planes of neighbors from detecting the fault.
- AMB and SMB: run control plane processes.

### Implementation

With IS-IS NSR, IS-IS real-time data is synchronized between the AMB and SMB. After an AMB/SMB switchover is performed on a device, the SMB takes over services from the AMB, and neighbors are unaware of the local fault. After the switchover, the new AMB restores IS-IS immediately based on the synchronized IS-IS real-time data. Therefore, neighbors are unaware of the switchover as well. IS-IS NSR requires synchronization of the following data:

- All configuration data, such as information about neighbors, timer parameters, and process configurations.
- Dynamic data, such as the interface parameters and state, and information about neighbors and the link state database (LSDB).



For details about NSR, see *Feature Description - Reliability*.

## Usage Scenario

NSR minimizes the impact of control plane faults and prevents route flapping on networks that require high reliability.

## Benefits

NSR improves network reliability and ensures uninterrupted traffic forwarding.

## IS-IS for IPv6

Standard protocols released by the IETF defines two new TLVs that can support IPv6 routes and a new Network Layer Protocol Identifier (NLPID), which ensures that IS-IS can process and calculate IPv6 routes.

The two new TLVs are as follows:

- IPv6 Reachability

The IPv6 Reachability TLV indicates the reachability of a network by specifying the route prefix and metric. The type value is 236 (0xEC).

- IPv6 Interface Address

The IPv6 Interface Address TLV is similar to the IP interface address TLV of IPv4 in function, except that it changes the original 32-bit IPv4 address to a 128-bit IPv6 address. The type value is 232 (0xE8).

The NLPID is an 8-bit field that identifies network layer protocol packets. The NLPID of IPv6 is 142 (0x8E). If an IS-IS router supports IPv6, it advertises routing information through the NLPID value.

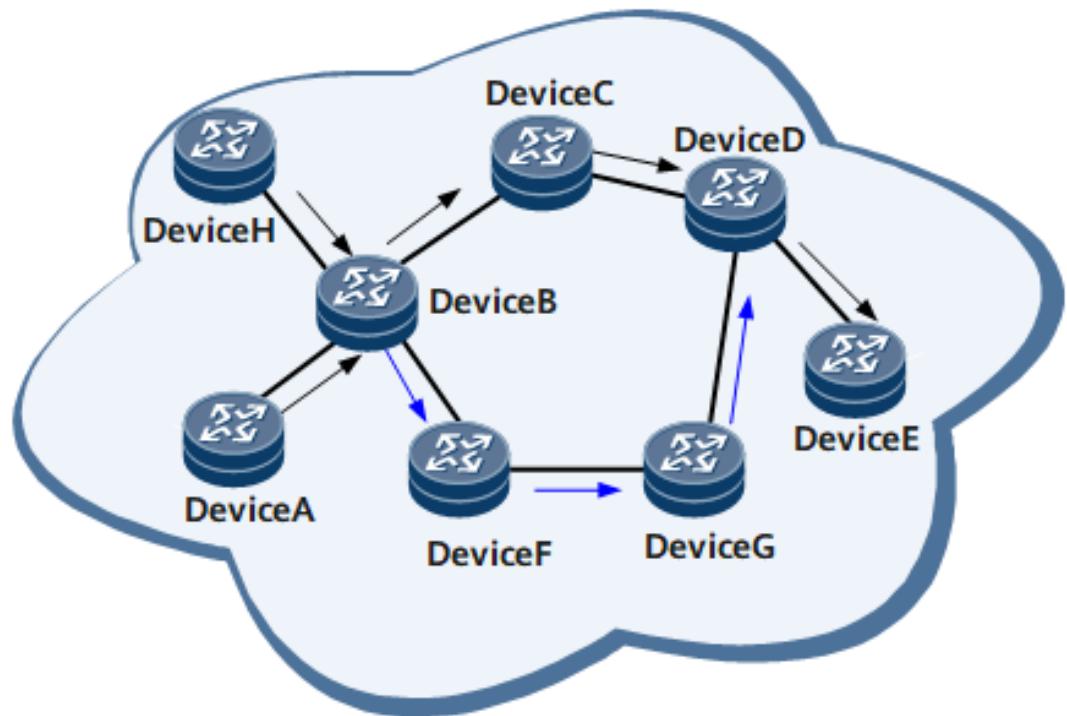
## IS-IS TE

IS-IS Traffic Engineering (TE) allows MPLS to set up and maintain TE constraint-based routed label switched paths (CR-LSPs).

To establish CR-LSPs, MPLS needs to learn the traffic attributes of all the links in the local area. MPLS can acquire the TE information of the links through IS-IS.

Traditional routers select the shortest path as the primary route regardless of other factors, such as bandwidth, even when the path is congested.

Figure 1-241 Networking with IS-IS routing defects



On the network shown in [Figure 1-241](#), all the links have the same metric (10). The shortest path from DeviceA/DeviceH to DeviceE is DeviceA/DeviceH → DeviceB → DeviceC → DeviceD → DeviceE. Data is forwarded along this shortest path. Therefore, the link DeviceA (DeviceH) → DeviceB → DeviceC → DeviceD → DeviceE may be congested whereas the link DeviceA/DeviceH → DeviceB → DeviceF → DeviceG → DeviceD → DeviceE is idle.

To solve the preceding problem, you can adjust the link metric. For example, based on topology analysis, you can adjust the metric of the link DeviceB → DeviceC to 30. In this manner, traffic can be diverted to the link DeviceA/DeviceH → DeviceB → DeviceF → DeviceG → DeviceD → DeviceE.

This method eliminates the congestion on the link DeviceA/DeviceH → DeviceB → DeviceC → DeviceD → DeviceE; however, the other link DeviceA/DeviceH → DeviceB → DeviceF → DeviceG → DeviceD → DeviceE may be congested. In addition, on a network with complex topologies, it is difficult to adjust the metric because the change in the metric of one link may affect multiple routes.

As an overlay model, MPLS can set up a virtual topology over the physical network topology and map traffic to the virtual topology, effectively combining MPLS and TE technology into MPLS TE.

MPLS TE has advantages in solving the problem of network congestion. Through MPLS TE, carriers can precisely control the path through which traffic passes, thus avoiding congested nodes. In addition, MPLS TE reserves resources during tunnel establishment to ensure service quality.

To ensure service continuity, MPLS TE introduces the path backup and fast reroute (FRR) mechanisms to switch traffic in time when a link is faulty. MPLS TE allows

service providers (SPs) to fully utilize existing network resources to provide diversified services. In addition, network resources can be optimized for scientific network management.

To accomplish the preceding tasks, MPLS TE needs to learn TE information about all devices on the network. However, MPLS TE lacks a mechanism in which each device floods its TE information throughout the entire network for TE information synchronization. However, IS-IS does provide such a mechanism. Therefore, MPLS TE can advertise and synchronize TE information with the help of IS-IS. To support MPLS TE, IS-IS needs to be extended.

In brief, IS-IS TE collects TE information on IS-IS networks and then transmits the TE information to the Constrained Shortest Path First (CSPF) module.

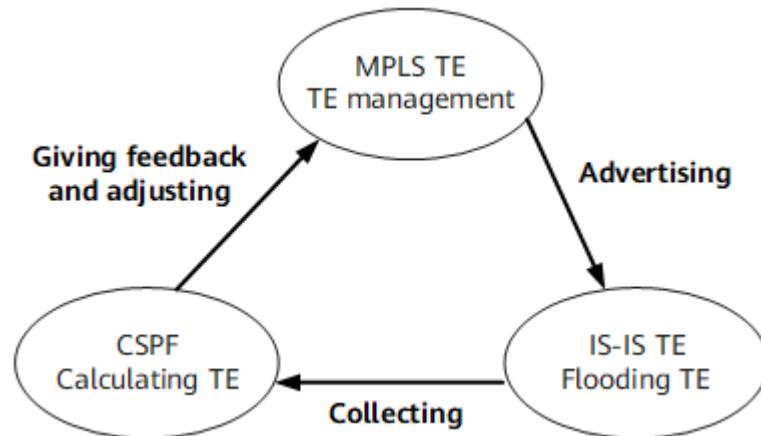
 NOTE

Currently, IS-IS TE supports only IPv4.

## Basic Principles

IS-IS TE is an extension of IS-IS intended to support MPLS TE. As defined in standard protocols, IS-IS TE uses LSPs to carry TE information to help MPLS implement the flooding, synchronization, and resolution of TE information. Then, IS-IS TE transmits the resolved TE information to the CSPF module. In MPLS TE, IS-IS TE plays the role of a porter. [Figure 1-242](#) illustrates the relationships between IS-IS TE, MPLS TE, and CSPF.

**Figure 1-242** Outline of relationships between MPLS TE, CSPF, and IS-IS TE



To carry TE information in LSPs, IS-IS TE defines the following TLVs in standard protocols:

- Extended IS reachability TLV

This TLV replaces the IS reachability TLV and extends the TLV format using sub-TLVs. The implementation of the sub-TLVs in the TLV is the same as that of TLVs in LSPs. These sub-TLVs are used to carry TE information configured on physical interfaces.

 NOTE

All sub-TLVs defined in standard protocols are supported.

**Table 1-86** Sub-TLVs defined in Extended IS reachability TLV

| Name                                      | Type | Length (Byte) | Value                                     |
|-------------------------------------------|------|---------------|-------------------------------------------|
| Administrative Group                      | 3    | 4             | Administrative group                      |
| IPv4 Interface Address                    | 6    | 4             | IPv4 address of a local interface         |
| IPv4 Neighbour Address                    | 8    | 4             | IPv4 address of a neighbor's interface    |
| Maximum Link Bandwidth                    | 9    | 4             | Maximum link bandwidth                    |
| Maximum Reserved Link Bandwidth           | 10   | 4             | Maximum reserved link bandwidth           |
| Unreserved Bandwidth                      | 11   | 32            | Unreserved bandwidth                      |
| Traffic Engineering Default Metric        | 18   | 3             | Default metric of TE                      |
| Bandwidth Constraints sub-TLV             | 22   | 36            | Sub-TLV of the bandwidth constraint.      |
| Unidirectional Link Delay Sub-TLV         | 33   | 4             | Average unidirectional link delay         |
| Min/Max Unidirectional Link Delay Sub-TLV | 34   | 8             | Minimum/Maximum unidirectional link delay |
| Unidirectional Delay Variation Sub-TLV    | 35   | 4             | Unidirectional link delay jitter          |

- Traffic Engineering router ID TLV  
The type of this TLV is 134, and this TLV carries a 4-byte router ID (MPLS LSR-ID). In MPLS TE, each device has a unique router ID.
- Extended IP reachability TLV  
This TLV replaces the IP reachability TLV and carries routing information. It extends the length of the route cost field to 4 bytes and carries sub-TLVs.

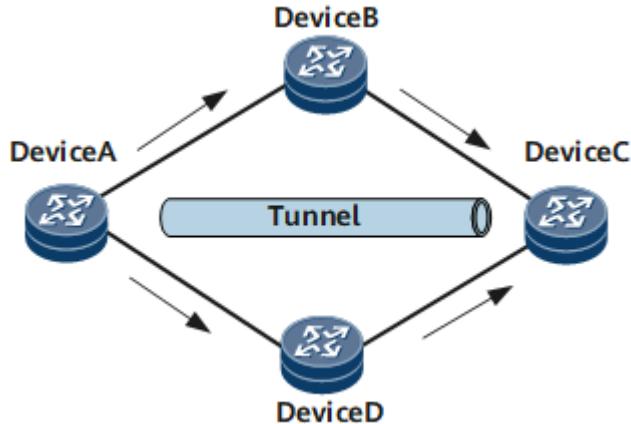
IS-IS TE consists of two procedures:

- Responding to MPLS TE configurations  
IS-IS TE functions only after MPLS TE is enabled.  
It updates the TE information in IS-IS LSPs based on MPLS TE configurations.  
It transmits MPLS TE configurations to the CSPF module.
- Processing TE information in LSPs  
It extracts TE information from IS-IS LSPs and transmits the TE information to the CSPF module.

## Usage Scenario

IS-IS TE helps MPLS TE set up TE tunnels. In [Figure 1-243](#), a TE tunnel is set up between Device A and Device C.

**Figure 1-243** Networking for IS-IS TE



The configuration requirements are as follows:

- MPLS TE and MPLS TE CSPF are enabled on Device A.
- MPLS TE is enabled on Device B, Device C, and Device D.
- IS-IS and IS-IS TE are enabled on Device A, Device B, Device C, and Device D.

After the configurations are complete, IS-IS on Device A, Device B, Device C, and Device D sends LSPs carrying TE information configured on each device. Device A obtains the MPLS TE configurations of DeviceB, DeviceC, and DeviceD from the received LSPs. In this way, Device A obtains the TE information of the entire network. The CSPF module can use the information to calculate the path required by the tunnel.

## IS-IS Wide Metric

In the earlier ISO 10589, the largest metric of an interface is 63. TLV type 128 and TLV type 130 contain information about routes, and TLV type 2 contains information about IS-IS neighbors. However, on large-scale networks, the metric range cannot meet the requirements. Moreover, IS-IS TE needs to be configured. Therefore, the wide metric was introduced.

As defined in standard protocols, with IS-IS wide metric, the largest metric of an interface is extended to 16777215, and the largest metric of a route is 4261412864.

After IS-IS wide metric is enabled, TLV type 135 contains information about routes; TLV type 22 contains information about IS-IS neighbors.

- The following lists the TLVs used in narrow mode:
  - IP Internal Reachability TLV: carries routes within an area.
  - IP External Reachability TLV: carries routes outside an area.

- IS Neighbors TLV: carries information about neighbors.
- The following lists the TLVs used in wide mode:
  - Extended IP Reachability TLV: replaces the earlier IP Reachability TLV and carries information about routes. This TLV expands the range of the route cost to 4 bytes and carries sub-TLVs.
  - IS Extended Neighbors TLV: carries information about neighbors.

 **NOTE**

The metric style can be set to narrow, narrow-compatible, compatible, wide-compatible, or wide mode. **Table 1-87** shows which metric styles are carried in received and sent packets. A device can calculate routes only when it can receive, send, and process corresponding TLVs. Therefore, to ensure correct data forwarding on a network, the proper metric style must be configured for each device on the network.

**Table 1-87** Metric style carried in received and sent under different metric style configurations

| Configured Metric Style | Metric Style Carried in Received Packets | Metric Style Carried in Sent Packets |
|-------------------------|------------------------------------------|--------------------------------------|
| Narrow                  | Narrow                                   | Narrow                               |
| Narrow-compatible       | Narrow and wide                          | Narrow                               |
| Compatible              | Narrow and wide                          | Narrow and wide                      |
| Wide-compatible         | Narrow and wide                          | Wide                                 |
| Wide                    | Wide                                     | Wide                                 |

When the metric style is set to compatible, IS-IS sends the information both in narrow and wide modes.

## Process

**NOTICE**

Once the metric style is changed, the IS-IS process restarts.

- If the metric style carried in sent packets is changed from narrow to wide:  
The information previously carried by TLV type 128, TLV type 130, and TLV type 2 is now carried by TLV type 135 and TLV type 22.
- If the metric style carried in sent packets is changed from wide to narrow:  
The information previously carried by TLV type 135 and TLV type 22 is now carried by TLV type 128, TLV type 130, and TLV type 2.
- If the metric style carried in sent packets is changed from narrow or wide to narrow and wide:  
The information previously carried in narrow or wide mode is now carried by TLV type 128, TLV type 130, TLV type 2, TLV type 135, and TLV type 22.

## Usage Scenario

IS-IS wide metric is used to support IS-IS TE, and the metric style needs to be set to wide, compatible or wide compatible.

## BFD for IS-IS

In most cases, the interval at which Hello packets are sent is 10s, and the IS-IS neighbor holding time (the timeout period of a neighbor relationship) is three times the interval. If a device does not receive a Hello packet from its neighbor within the holding time, the device terminates the neighbor relationship.

A device can detect neighbor faults at the second level only. As a result, link faults on a high-speed network may cause a large number of packets to be discarded.

BFD, which can be used to detect link faults on lightly loaded networks at the millisecond level, is introduced to resolve the preceding issue. With BFD, two systems periodically send BFD packets to each other. If a system does not receive BFD packets from the other end within a specified period, the system considers the bidirectional link between them Down.

BFD is classified into the following modes:

- Static BFD

In static BFD mode, BFD session parameters (including local and remote discriminators) are set using commands, and requests must be delivered manually to establish BFD sessions.

- Dynamic BFD

In dynamic BFD mode, the establishment of BFD sessions is triggered by routing protocols.

BFD for IS-IS enables BFD sessions to be dynamically established. After detecting a fault, BFD notifies IS-IS of the fault. IS-IS sets the neighbor status to Down, quickly updates link state protocol data units (LSPs), and performs the partial route calculation (PRC). BFD for IS-IS implements fast IS-IS route convergence.

### NOTE

Instead of replacing the Hello mechanism of IS-IS, BFD works with IS-IS to rapidly detect the faults that occur on neighboring devices or links.

## BFD Session Establishment and Deletion

- Conditions for establishing a BFD session
  - Global BFD is enabled on each device, and BFD is enabled on a specified interface or process.
  - IS-IS is configured on each device and enabled on interfaces.
  - Neighbors are Up, and a designated intermediate system (DIS) has been elected on a broadcast network.
- Process of establishing a BFD session
  - P2P network

After the conditions for establishing BFD sessions are met, IS-IS instructs the BFD module to establish a BFD session and negotiate BFD parameters between neighbors.

- Broadcast network

After the conditions for establishing BFD sessions are met and the DIS is elected, IS-IS instructs BFD to establish a BFD session and negotiate BFD parameters between the DIS and each device. No BFD sessions are established between non-DISs.

On broadcast networks, devices (including non-DIS devices) of the same level on a network segment can establish adjacencies. In BFD for IS-IS, however, BFD sessions are established only between the DIS and non-DISs. On P2P networks, BFD sessions are directly established between neighbors.

If a Level-1-2 neighbor relationship is set up between the devices on both ends of a link, the following situations occur:

- On a broadcast network, IS-IS sets up a Level-1 BFD session and a Level-2 BFD session.
- On a P2P network, IS-IS sets up only one BFD session.

- Process of tearing down a BFD session

- P2P network

If the neighbor relationship established between P2P IS-IS interfaces is not Up, IS-IS tears down the BFD session.

- Broadcast network

If the neighbor relationship established between broadcast IS-IS interfaces is not Up or the DIS is reelected on the broadcast network, IS-IS tears down the BFD session.

If the configurations of dynamic BFD sessions are deleted or BFD for IS-IS is disabled from an interface, all Up BFD sessions established between the interface and its neighbors are deleted. If the interface is a DIS and the DIS is Up, all BFD sessions established between the interface and its neighbors are deleted.

If BFD is disabled from an IS-IS process, BFD sessions are deleted from the process.

 NOTE

BFD detects only the one-hop link between IS-IS neighbors because IS-IS establishes only one-hop neighbor relationships.

- Response to the Down event of a BFD session

When BFD detects a link failure, it generates a Down event and informs IS-IS. IS-IS then suppresses neighbor relationships and recalculates routes. This process speeds up network convergence.

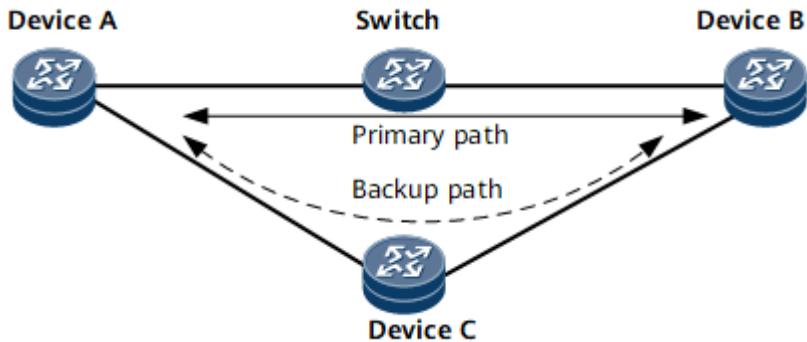
## Usage Scenario

**NOTICE**

Dynamic BFD needs to be configured based on the actual network. If the time parameters are not configured correctly, network flapping may occur.

BFD for IS-IS speeds up route convergence through rapid link failure detection. The following is a networking example for BFD for IS-IS.

Figure 1-244 BFD for IS-IS



The configuration requirements are as follows:

- Basic IS-IS functions are configured on each device shown in [Figure 1-244](#).
- Global BFD is enabled.
- BFD for IS-IS is enabled on Device A and Device B.

If the link between Device A and Device B fails, BFD can rapidly detect the fault and report it to IS-IS. IS-IS sets the neighbor status to Down to trigger an IS-IS topology calculation. IS-IS also updates LSPs so that Device C can promptly receive the updated LSPs from Device B, which accelerates network topology convergence.

## IS-IS Auto FRR

### Background

As networks develop, services such as Voice over IP (VoIP) and online video services require high-quality and real-time transmission. However, if a link fails, IS-IS must complete the following procedure before switching traffic to a new link: detect the fault, update LSPs, flood LSPs, calculate routes, and deliver route entries to the FIB. This is a lengthy process, and the associated traffic interruption is often longer than users can tolerate. As a result, real-time transmission requirements cannot be met.

IS-IS Auto FRR (Fast reroute) is dynamic IP FRR. An IGP pre-calculates a backup path based on the LSDBs on the entire network and saves the backup path in the forwarding table for traffic protection in the case of a fault. Because IP FRR helps implement fast route convergence, it becomes increasingly popular with carriers.

Currently, main FRR technologies include LFA, U-turn, Not-Via, TI-LFA, Remote-LFA, and MRT, among which IS-IS only supports LFA, TI-LFA, and Remote-LFA.

### Related Concepts

#### LFA

LFA is an IP FRR technology that calculates the shortest path from the neighbor that can provide a backup link to the destination node based on the Shortest Path First (SPF) algorithm. Then, LFA calculates a loop-free backup link with the smallest cost based on the inequality:

Distance<sub>opt</sub> (N, D) < Distance<sub>opt</sub> (N, S) + Distance<sub>opt</sub> (S, D), in which S, D, and N indicate the source node, destination node, and a node on the backup link, respectively, and Distance<sub>opt</sub> (X, Y) indicates the shortest distance from node X to node Y.

### P space

The P space is a set of nodes reachable from the source node of the primary link using the shortest path tree (SPT) rooted at the source node without traversing the primary link.

### Extended P space

Extended P space consists of the nodes through which the SPTs with neighbors of a primary link's source node as the root are reachable without passing through the primary link.

### Q space

The Q space is a set of nodes reachable from the destination node using the reverse SPT rooted at the destination node without traversing the primary link.

### PQ node

The nodes in both the extended P space and Q space are PQ nodes and are used as the destinations of protection tunnels in remote LFA.

### Remote LFA

LFA FRR cannot be used to calculate backup links on some large-scale networks, especially on ring networks. Remote LFA FRR addresses this problem by calculating a PQ node and establishing a tunnel between the source node and the PQ node as a backup path. If the primary link fails, traffic can be automatically switched to the backup path, which improves network reliability.

#### NOTE

When calculating an RLFA FRR backup path, a Huawei device calculates the extended P space by default.

### TI-LFA

In some LFA FRR and RLFA scenarios, the extended P space and Q space neither intersect nor have direct neighbors. Consequently, no backup path can be computed, failing to meet reliability requirements. TI-LFA solves this problem by computing the extended P space, Q space, and post-convergence SPT based on the protected path, computing a scenario-specific repair list, and establishing an SR tunnel from the source node to a P node and then to a Q node to offer alternate next hop protection. If the protected link fails, traffic is automatically switched to the backup path, improving network reliability.

#### NOTE

When computing a TI-LFA FRR backup path, Huawei devices compute the extended P space by default.

For more information about TI-LFA, see TI-LFA FRR.

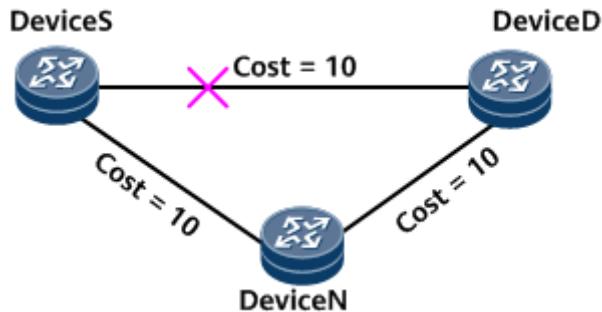
## IS-IS LFA Auto FRR

IS-IS LFA Auto FRR protects against both link and node-and-link failures.

- Link protection: Link protection applies to traffic transmitted over specified links.

In the example network shown in [Figure 1-245](#), traffic flows from DeviceS to DeviceD, and the link cost meets the preceding link protection inequality. If the primary link (DeviceS -> DeviceD) fails, DeviceS switches the traffic to the backup link (DeviceS -> DeviceN -> DeviceD), minimizing traffic loss.

[Figure 1-245](#) Networking for IS-IS LFA Auto FRR link protection



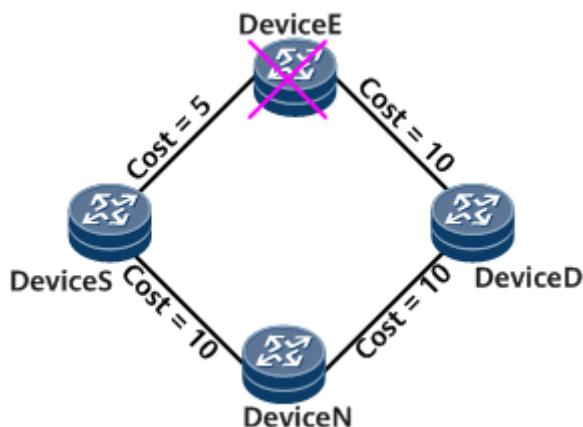
- Node-and-link protection: Node-and-link protection applies to traffic transmitted over specified nodes or links. [Figure 1-246](#) illustrates a network with node-and-link protection. Node-and-link protection takes precedence over link protection.

Node-and-link protection takes effect when the following conditions are met:

- The link cost satisfies the inequality:  $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, S) + \text{Distance}_{\text{opt}}(S, D)$ .
- The interface cost of the device satisfies the inequality:  $\text{Distance}_{\text{opt}}(N, D) < \text{Distance}_{\text{opt}}(N, E) + \text{Distance}_{\text{opt}}(E, D)$ .

S indicates the source node of traffic, E indicates the faulty node, N indicates the node on the backup link, and D indicates the destination node of traffic.

[Figure 1-246](#) Networking for IS-IS LFA Auto FRR node-and-link protection

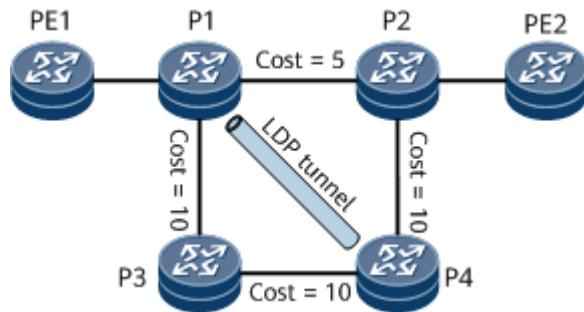


## IS-IS Remote LFA Auto FRR

Similar to IS-IS LFA Auto FRR, Remote LFA is also classified as link protection or node-and-link protection. The following example shows how Remote LFA works to protect against link failures:

In [Figure 1-247](#), traffic flows through PE1 -> P1 -> P2 -> PE2. To prevent traffic loss caused by a link fault between P1 and P2, Remote LFA calculates a PQ node (P4) and establishes an LDP tunnel between P1 and P4. If P1 detects a failure on the link between P1 and P2, P1 encapsulates packets into MPLS packets and forwards the MPLS packets to P4. After receiving the packets, P4 removes the MPLS label from them, searches its IP routing table for a next hop, and forwards the packets to the next hop. Finally, the packets reach their destination (PE2). This implements fast path switching and avoids traffic loss.

**Figure 1-247** Networking for Remote LFA



On the network shown in [Figure 1-247](#), PQ nodes are calculated as follows:

1. SPTs are calculated, with each of P1's neighbors (excluding the neighbors on the primary link) as the root. In this case, the neighbors PE1 and P3 are used as roots. For each SPT, an extended P space is composed of the root node and those nodes reachable through non-P1-P2 links. When PE1 is used as a root node for calculation, the extended P space {PE1, P1, P3} is obtained. When P3 is used as a root node for calculation, the extended P space {PE1, P1, P3, P4} is obtained. By combining these two extended P spaces, the final extended P space {PE1, P1, P3, P4} is obtained.
2. A reverse SPT with PE2 as the root is calculated. The Q space is {P2, PE2, P4}.
3. The node that exists both in the extended P space and Q space is the PQ node. In this example, P4 is the PQ node.

### NOTE

IPv6 IS-IS Remote LFA Auto FRR protects IPv6 traffic and uses IPv4 LDP LSPs. The principle of IPv6 IS-IS Remote LFA Auto FRR is similar to that of IPv4 IS-IS Remote LFA Auto FRR.

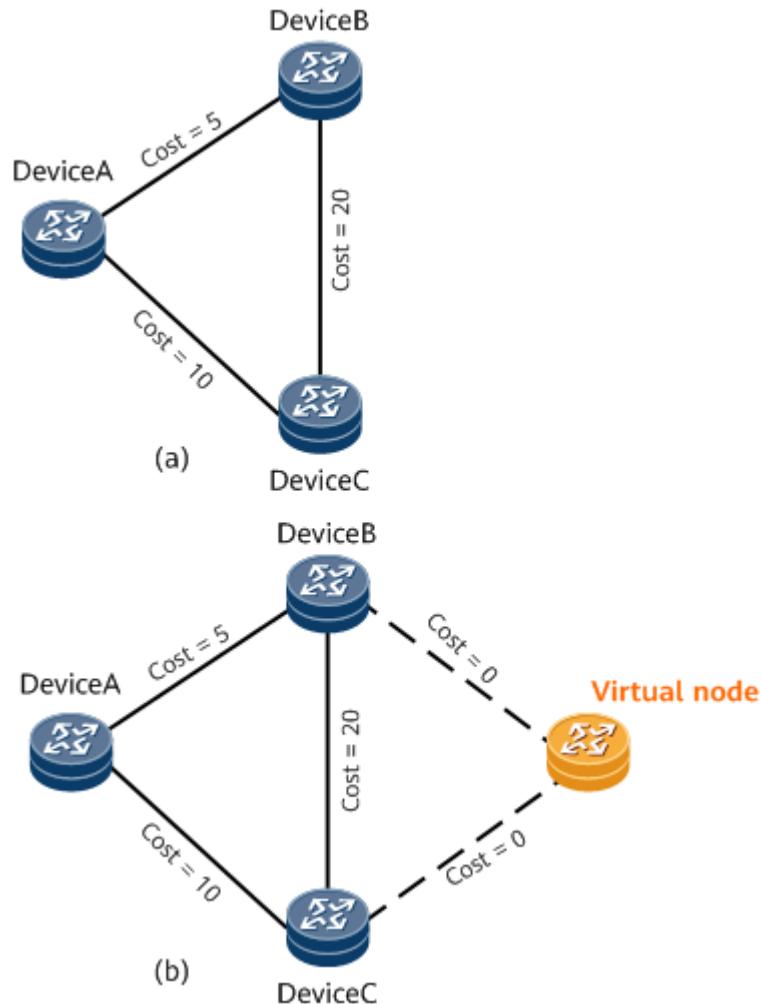
On a network with a large number of nodes, to ensure that RLFA/TI-LFA calculation can be completed as soon as possible, the elected P and Q nodes may not be optimal, but they comply with rules. This does not affect the protection effect.

## IS-IS FRR in the Scenario Where Multiple Nodes Advertise the Same Route

IS-IS LFA FRR uses the SPF algorithm to calculate the shortest path to the destination node, with each neighbor that provides a backup link as the root node.

The backup next hop is node-based, which applies to single-node routing scenarios, that is, scenarios where a route is received from only one node. With the diversification of networks, multi-node routing scenarios appear, where multiple nodes advertise the same route. Such scenarios do not meet single-node LFA conditions. As a result, the backup next hop cannot be calculated. IS-IS FRR for multi-node routing scenarios can address this problem by using a routing source to protect the primary routing source and improve network reliability.

**Figure 1-248** IS-IS FRR in the scenario where multiple nodes advertise the same route



In **Figure 1-248(a)**, the cost of the link between Device A and Device B is 5, whereas the cost of the link between Device A and Device C is 10. Both Device B and Device C advertise the route 10.1.1.0/24. IS-IS FRR is enabled on Device A. However, Device A cannot calculate the backup next hop of the route 10.1.1.0/24 because the LFA conditions in the scenario where each route is received from a single node cannot be met. To address this issue, IS-IS FRR for the scenario where multiple nodes advertise the same route can be used.

In **Figure 1-248(b)**, a virtual node is simulated between Device B and Device C and is connected to Device B and Device C. The cost of the link from Device B or Device C to the virtual node is 0, whereas the cost of the link from the virtual

node to Device B or Device C is the maximum value. After the virtual node advertises the route 10.1.1.0/24, Device A can use the LFA algorithm to calculate the backup next hop of the virtual node because the scenario where multiple nodes advertise the same route has been converted to the scenario where the route is received from only one node. Then the route 10.1.1.0/24 inherits the backup next hop from the virtual node. In this example, the primary link to the virtual node is the one from Device A to Device B, and the backup link is the one from Device A to Device C.

## IS-IS ECMP FRR

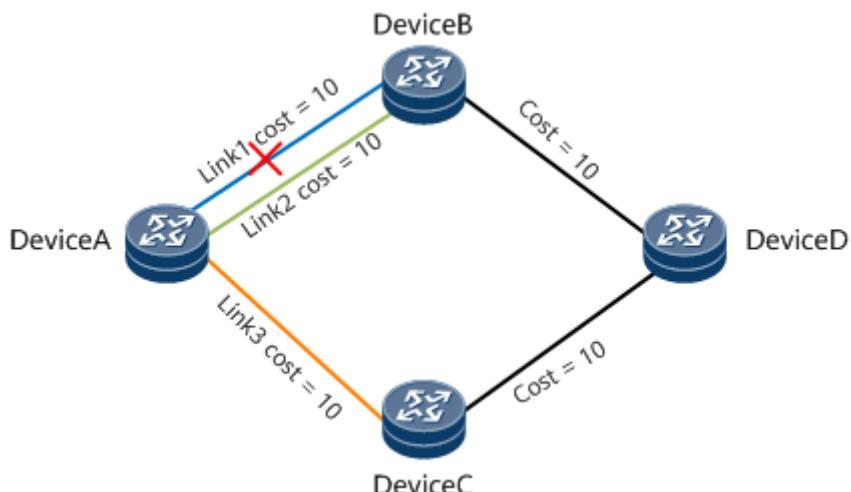
Equal cost multi path (ECMP) evenly balances traffic over multiple equal-cost paths to the same destination.

If the ECMP FRR function is not supported in ECMP scenarios, no backup next hop can be calculated for primary links.

IS-IS ECMP FRR is enabled by default, and a backup next hop is calculated separately for each primary link, which enhances reliability in ECMP scenarios. With ECMP FRR, IS-IS pre-calculates backup paths for load balancing links based on the LSDBs on the entire network. The backup paths are stored in the forwarding table and are used for traffic protection in the case of link failures.

- In **Figure 1-249**, traffic is forwarded from Device A to Device D and is balanced among Link 1, Link 2, and Link 3. Backup paths of the three links are calculated based on ECMP FRR. For example, the backup paths of Link 1, Link 2, and Link 3 are Link 3, Link 3, and Link 2, respectively.
  - If the ECMP FRR function is not enabled in the load balancing scenario and Link 1 fails, traffic over Link 1 is randomly switched to Link 2 or Link 3, which affects service traffic management.
  - If the ECMP FRR function is enabled in the load balancing scenario and Link 1 fails, traffic over Link 1 is switched to Link 3 according to FRR route selection rules, which enhances service traffic management.

**Figure 1-249** Flexible selection of backup paths through IS-IS ECMP FRR

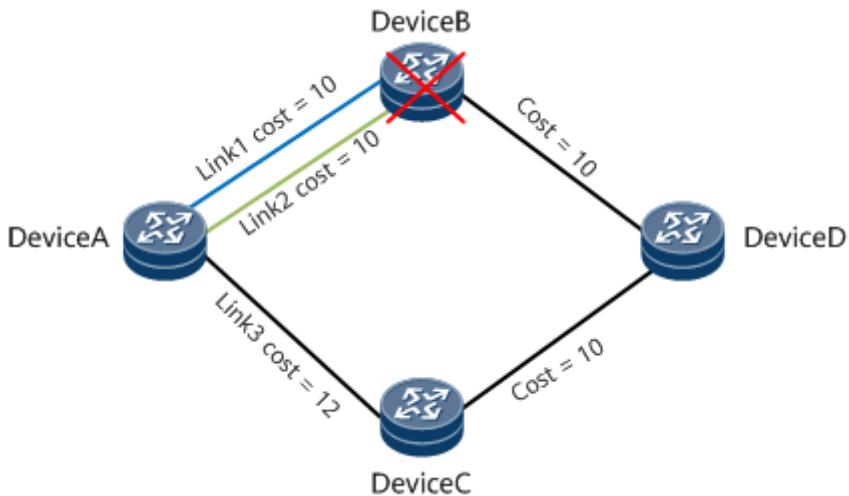


- In **Figure 1-250**, traffic is forwarded from Device A to Device D and is balanced between Link 1 and Link 2. Backup paths of the two links are

calculated based on ECMP FRR. For example, the backup paths of Link 1 and Link 2 are both Link 3.

- If the ECMP FRR function is not enabled in the load balancing scenario and Device B fails, Link 1 and Link 2 fail accordingly, leading to a traffic interruption.
- If the ECMP FRR function is enabled in the load balancing scenario and Device B fails, Link 1 and Link 2 fail accordingly. However, traffic is switched to Link 3, which prevents the traffic interruption.

**Figure 1-250** Enhanced traffic protection through IS-IS ECMP FRR



## IS-IS SRLG FRR

A shared risk link group (SRLG) is a set of links that share a common physical resource, such as an optical fiber. These links share the same risk level. If one of the links fails, all the other links in the SRLG may also fail.

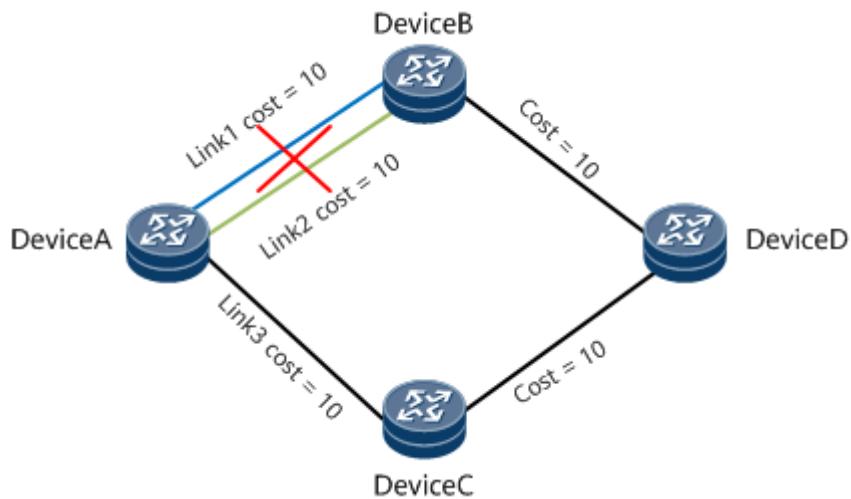
In [Figure 1-251](#), IS-IS SRLG FRR is configured, traffic between Device A and Device B is balanced by Link 1 and Link 2.

If IS-IS LFA Auto FRR is enabled, it implements protection for Link 1 and Link 2. That is, the two links back up each other.

- If Link 1 fails but Link 2 is normal, traffic is not interrupted after being switched to the backup link.
- If both Link 1 and Link 2 fail, traffic is interrupted after being switched to the backup link.

IS-IS SRLG FRR prevents service interruptions in the scenario where links have the same risk of failure. Add Link1 and Link2 to the same SRLG group so that the device preferentially selects a link in a different SRLG group when calculating a backup path, reducing the possibility of network traffic interruption.

Figure 1-251 Networking for IS-IS SRLG FRR



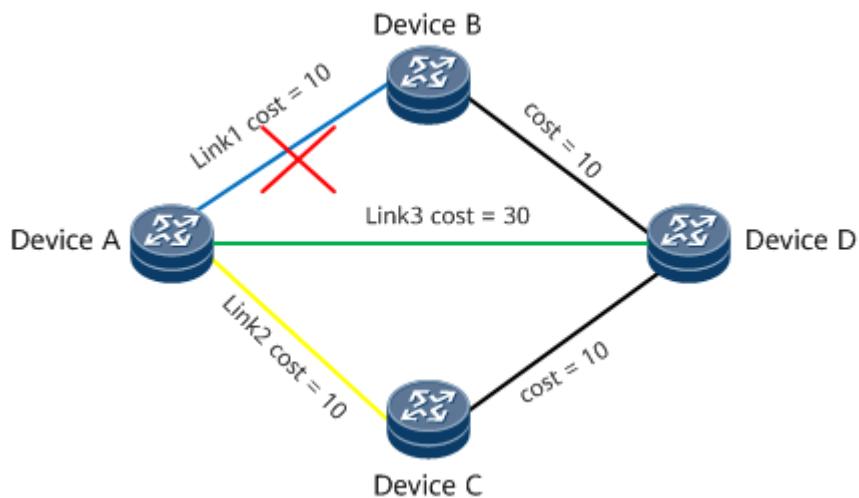
In an IS-IS remote SRLG FRR scenario shown in [Figure 1-252](#), traffic is forwarded from Device A to Device B.

In this case, Link1 is the primary path and Link2 is the backup path. If the link between Device C and Device D and Link1 are in the same SRLG:

- If Link1 fails but the link between Device C and Device D is normal, network traffic is switched to the backup path and can be forwarded normally.
- If both Link1 and the link between Device C and Device D fail, traffic is interrupted after being switched to the backup path.

If a remote link has the same risk of failure with a local link, configure IS-IS remote SRLG TI-LFA FRR so that the device preferentially selects a link that is not in an SRLG group with local links when calculating a backup path. As shown in Figure 8, Link3 is preferentially selected as the backup path, reducing the possibility of network traffic interruptions.

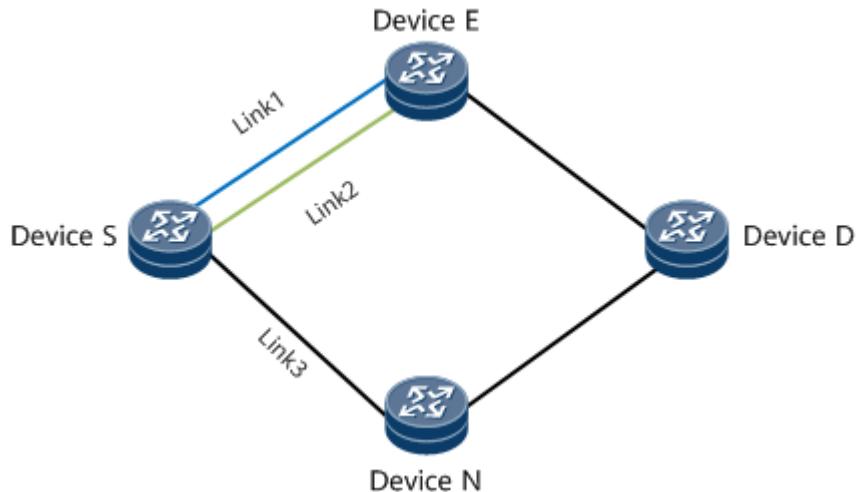
Figure 1-252 Networking diagram of IS-IS remote SRLG FRR



## Priority-based Selection of a Backup Path for IS-IS Auto FRR

When selecting a backup path for IS-IS Auto FRR, a node-protection solution is preferentially selected. However, in actual network operation, the solution of selecting a backup path for IS-IS Auto FRR may need to be changed due to factors such as interface data capabilities or link costs.

**Figure 1-253** Configuring the priority-based selection of a backup path for IS-IS Auto FRR



### Scenario 1

In [Figure 1-253](#), if the primary path is Link 1 (DeviceS -> DeviceE -> DeviceD), the backup path can be Link 2 (DeviceS -> DeviceE -> DeviceD) or Link 3 (DeviceS -> DeviceN-> DeviceD). The cost of Link 1, Link 2, and Link 3 are 10, 20, and 100, respectively. By default, Link 3 is preferentially selected as the backup path. To change the solution of selecting a backup path to smallest-cost path first, run the **tiebreaker lowest-cost preference [ level-1 | level-2 | level-1-2 ]** command. After the command is run, Link 2 is preferentially selected as the backup path.

### Scenario 2

In [Figure 1-253](#), if traffic is forwarded from Device S to Device E, it is balanced between Link 1 and Link 2 between Device S and Device E. Device S and Device N are connected through Link 3. The cost of Link 1, Link 2 and Link 3 are 10, 10, and 15 respectively. When selecting backup paths for load balancing links, Link 1 and Link 2 select each other as the backup path. To change the solution of selecting a backup path to non-load balancing path first, run the **tiebreaker non-ecmp preference [ level-1 | level-2 | level-1-2 ]** command. After the command is run, Link 3 is preferentially selected as the backup path.

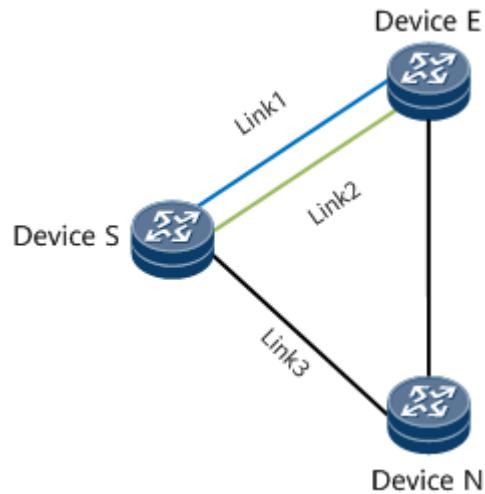
### Scenario 3

In [Figure 1-253](#), traffic is forwarded from Device A to Device D and is balanced between Link 1 and Link 2. Device S and Device N are connected through Link 3. The cost of Link 1, Link 2, and Link 3 are all 10. If IS-IS LFA Auto FRR is enabled, it implements protection for Link 1 and Link 2. That is, the two links back up each other. If Link 1 and Link 2 have the same risk of failure, you can run the

**tiebreaker srlg-disjoint preference preference [ level-1 | level-2 | level-1-2 ]** command to set the solution of selecting a backup path to SRLG-disjoint path first. In this case, the device preferentially selects Link 3 as the backup path. Before using this solution, run the **isis srlg srlg-value** command to add the links having the same risk of failure to an SRLG.

#### Scenario 4

**Figure 1-254** Selecting the link with the maximum cost as the backup path for IS-IS Auto FRR



On the network shown in [Figure 1-254](#), traffic is forwarded from Device S to Device E. The cost of Link 1 and Link 2 are 10 and 30, respectively, and the **isis peer hold-max-cost** command is run for Link 1. Device S and Device N are connected through Link 3, and Device N and Device E are directly connected. The cost of Link 3 and the link between Device N and Device E are both 10. In normal cases, the traffic from Device S to Device E is forwarded over Link 1, and the backup path is Link 3 (node protection is preferred). If Link 1 fails, the new primary path from Device S to Device E becomes Link 3, and the new backup path becomes Link 2. If Link 1 recovers after the **undo showdown** command is run, Link 1 advertises the maximum cost. In this case, to allow Link 1 to be selected as the backup path, run the **tiebreaker hold-max-cost preference preference [ level-1 | level-2 | level-1-2 ]** command.

## IS-IS Authentication

### Background

As the Internet develops, more data, voice, and video information are exchanged over the Internet. New services, such as e-commerce, online conferencing and auctions, video on demand, and distance learning, emerge gradually. The new services have high requirements for network security. Carriers need to prevent data packets from being illegally obtained or modified by attackers or unauthorized users. IS-IS authentication applies to the area or interface where packets need to be protected. Using IS-IS authentication enhances system security and helps carriers provide safe network services.

## Related Concepts

### Authentication Classification

Based on packet types, the authentication is classified as follows:

- Interface authentication: is configured in the interface view to authenticate Level-1 and Level-2 IS-to-IS Hello PDUs (IIHs).
- Area authentication: is configured in the IS-IS process view to authenticate Level-1 CSNPs, PSNPs, and LSPs.
- Routing domain authentication: is configured in the IS-IS process view to authenticate Level-2 CSNPs, PSNPs, and LSPs.

Based on the authentication modes of packets, the authentication is classified into the following types:

- Simple authentication: The authenticated party directly adds the configured password to packets for authentication. This authentication mode provides the lowest password security.
- MD5 authentication: uses the MD5 algorithm to encrypt a password before adding the password to the packet, which improves password security. For the sake of security, using the HMAC-SHA256 algorithm rather than the MD5 algorithm is recommended.
- Keychain authentication: further improves network security with a configurable key chain that changes with time.
- HMAC-SHA256 authentication: uses the HMAC-SHA256 algorithm to encrypt a password before adding the password to the packet, which improves password security.

## Implementation

IS-IS authentication encrypts IS-IS packets by adding the authentication field to packets to ensure network security. After receiving IS-IS packets from a remote router, a local router discards the packets if the authentication passwords in the packets are different from the locally configured one. This mechanism protects the local router.

IS-IS provides a type-length-value (TLV) to carry authentication information. The TLV components are as follows:

- Type: indicates the type of a packet, which is 1 byte. The value defined by ISO is 10, whereas the value defined by IP is 133.
- Length: indicates the length of the authentication TLV, which is 1 byte.
- Value: indicates the authentication information, including authentication type and authenticated password, which ranges from 1 to 254 bytes. The authentication type is 1 byte:
  - 0: reserved
  - 1: simple authentication
  - 3: general authentication, and only HMAC-SHA256 authentication currently
  - 54: MD5 authentication

- 255: private authentication

### Interface Authentication

Authentication passwords for IIHs are saved on interfaces. The interfaces send authentication packets with the authentication TLV. Interconnected router interfaces must be configured with the same password.

### Area Authentication

Every router in an IS-IS area must use the same authentication mode and have the same key chain.

### Routing Domain Authentication

Every Level-2 or Level-1-2 router in an IS-IS area must use the same authentication mode and have the same key chain.

For area authentication and routing domain authentication, you can set a router to authenticate SNPs and LSPs separately in the following ways:

- A router sends LSPs and SNPs that carry the authentication TLV and verifies the authentication information of the LSPs and SNPs it receives.
- A router sends LSPs that carry the authentication TLV and verifies the authentication information of the LSPs it receives. The router sends SNPs that carry the authentication TLV and does not verify the authentication information of the SNPs it receives.
- A router sends LSPs that carry the authentication TLV and verifies the authentication information of the LSPs it receives. The router sends SNPs without the authentication TLV and does not verify the authentication information of the SNPs it receives.
- A router sends LSPs and SNPs that carry the authentication TLV but does not verify the authentication information of the LSPs and SNPs it receives.

## IS-IS Purge Source Tracing

### Context

If network-wide IS-IS LSP deletion causes network instability, source tracing must be implemented as soon as possible to locate and isolate the fault source. However, IS-IS itself does not support source tracing. A conventional solution is to isolate nodes one by one until the fault source is located, but the process is complex and time-consuming and may compromise network services. To address this problem, enable IS-IS purge source tracing.

IS-IS purge source tracing is a Huawei proprietary protocol.

### Related Concepts

- PS-PDU: packets that carry information about the node that floods IS-IS purge LSPs.
- CAP-PDU: packets used to negotiate the IS-IS purge source tracing capability between IS-IS neighbors.
- IS-IS purge source tracing port: UDP port number used to send and receive IS-IS purge source tracing packets. This UDP port number is configurable.

## Fundamentals

IS-IS purge LSPs do not carry source information. If a device fails on the network, a large number of purge LSPs are flooded. Without a source tracing mechanism, nodes need to be isolated one by one until the faulty node is located, which is labor-intensive and time-consuming. IS-IS purge LSPs will trigger route flapping on the network, or even routes become unavailable. In this case, the device that floods the purge LSPs must be located and isolated immediately.

A solution that can meet the following requirements is required:

- 1. Information about the source that flooded the purge LSPs can be obtained when network routes are unreachable.
- 2. The method used to obtain source information must apply to all devices on the network and support incremental deployment, without compromising routing capabilities.

For requirement 1, IS-IS purge source tracing uses UDP to send and receive source tracing packets. These packets carry IS-IS LSP information purged by the faulty device and are flooded hop by hop along the IS-IS neighbor topology. After IS-IS purge source tracing packets are flooded, you can log in to any device that supports IS-IS purge source tracing to view information about the device that flooded the purge LSPs. This helps you quickly locate and isolate the faulty node.

For requirement 2, IS-IS purge source tracing forwards packets along UDP channels that are independent of the channels used to transmit IS-IS packets. In addition, source tracing does not affect the devices with the related UDP port disabled.

## Capability Negotiation

Source tracing packets are transmitted over UDP. Devices listen for the UDP port and use it to send and receive source tracing packets. If a source tracing-capable device sends source tracing packets to a device that is source tracing-incapable, the former may be incorrectly identified as an attacker. Therefore, the source tracing capability needs to be negotiated between devices so that source tracing packets are exchanged between only source tracing-capable devices. In addition, source tracing capability negotiation is also required to enable a source tracing-capable device to send source tracing information on behalf of a source tracing-incapable device.

Source tracing capability negotiation depends on IS-IS neighbor relationships. Specifically, after an IS-IS neighbor relationship is established, the local device initiates source tracing capability negotiation based on the IP address of the neighbor.

## PS-PDU Generation

If a fault source purges an LSP, it generates and floods a PS-PDU to all its source tracing neighbors.

If a device receives a purge LSP from a source tracing-incapable neighbor, the device generates and floods a PS-PDU to all its neighbors. If a device receives the same purge LSP (with the same LSP ID and sequence number) from more than one source tracing-incapable neighbor, the device generates only one PS-PDU.

PS-PDU flooding is similar to IS-IS LSP flooding.

## Security Concern

A UDP port is used to send and receive source tracing packets. Therefore, the security of the port must be taken into consideration.

The source tracing protocol inevitably increases packet receiving and sending workload and intensifies bandwidth pressure. To minimize its impact on other protocols, the number of source tracing packets must be controlled.

- Authentication

Source tracing is embedded in the IGP, inherits existing configuration parameters of the IGP, and uses authentication parameters of the IGP to authenticate packets.

- GTSM

GTSM is a security mechanism that checks whether the time to live (TTL) value in each received IP packet header is within a pre-defined range.

Source tracing packets can only be flooded as far as one hop. Therefore, GTSM can be used to check such packets by default. When a device sends a packet, it sets the TTL of the packet to 255. If the TTL is not 254 when the packet is received, the packet will be discarded.

- CPU-CAR

The NP module on interface boards can check the packets to be sent to the CPU for processing and prevent the main control board from being overloaded by a large number of packets that are sent to the CPU.

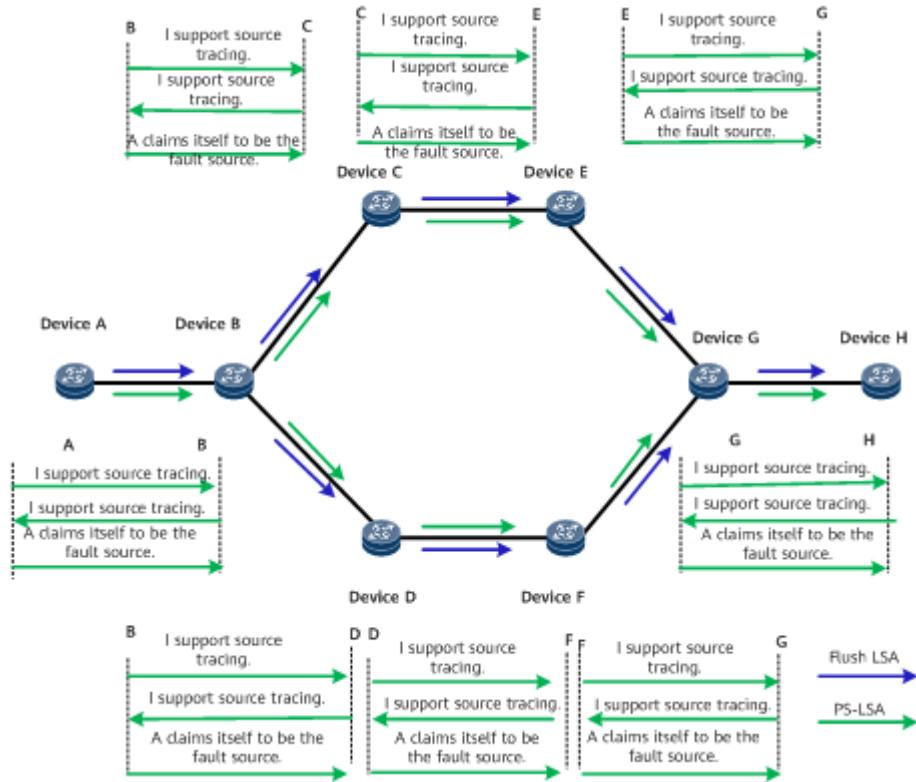
The source tracing protocol needs to apply for an independent CAR channel and has small CAR values configured.

## Typical Scenarios

### Scenario where all nodes support source tracing

Assume that all nodes on the network support source tracing and DeviceA is the fault source. In this scenario, the fault source can be accurately located. [Figure 1-255](#) shows the networking.

**Figure 1-255 Scenario where all nodes support source tracing**



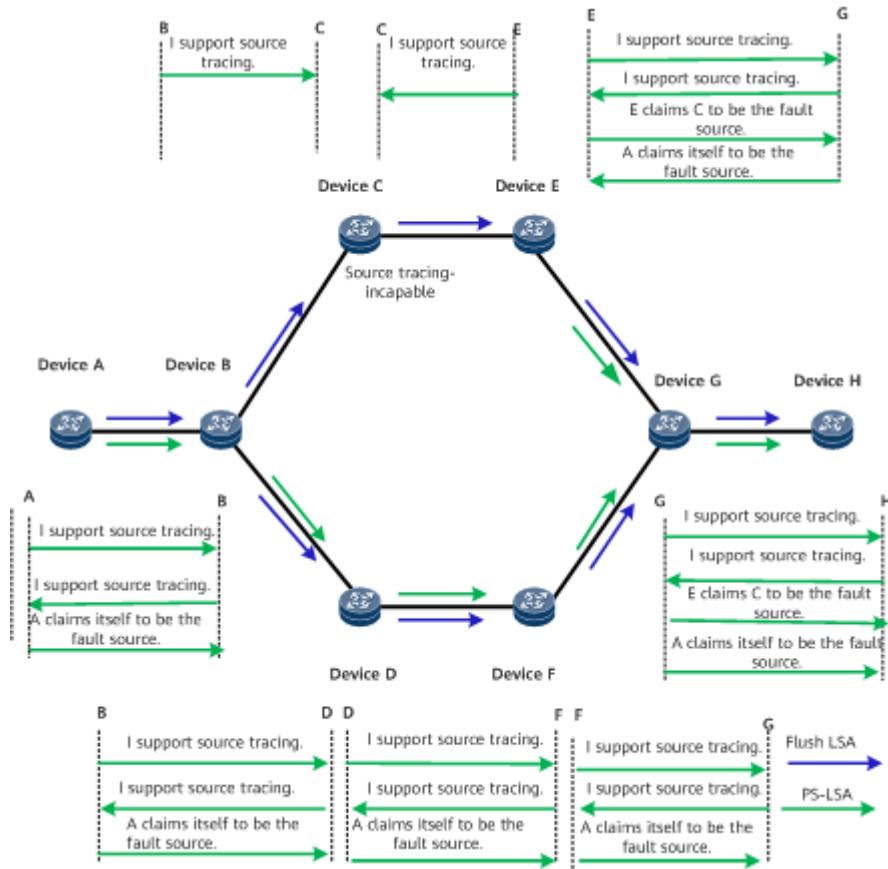
All nodes on the network support source tracing and DeviceA is the fault source.

When DeviceA purges an IGP packet, it floods a source tracing packet that carries DeviceA information and brief information about the IGP packet. Then the source tracing packet is flooded on the network hop by hop. After the fault occurs, maintenance personnel can log in to any node on the network to locate DeviceA, which keeps sending purge LSPs, and isolate DeviceA from the network.

#### Scenario where source tracing-incapable nodes are not isolated from source tracing-capable nodes

All nodes on the network except DeviceC support source tracing, and DeviceA is the faulty source. In this scenario, PS-PDUs can be flooded on the entire network, and the fault source can be accurately located. [Figure 1-256](#) shows the networking.

**Figure 1-256 Scenario where source tracing-incapable nodes are not isolated from source tracing-capable nodes**



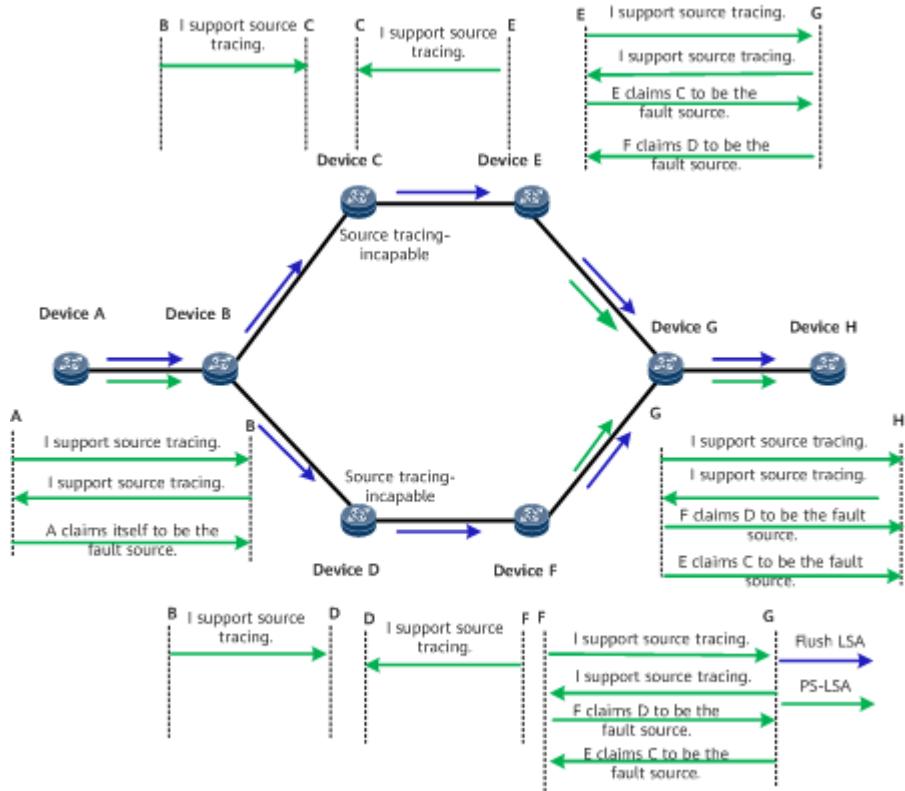
When DeviceA purges an IGP packet, it floods a source tracing packet that carries DeviceA information and brief information about the IGP packet. Then the source tracing packet is flooded on the network hop by hop. When DeviceB and DeviceE negotiate the source tracing capability with DeviceC, they find that DeviceC does not support source tracing. After DeviceB receives the PS-PDU from DeviceA, DeviceB sends the packet to DeviceD, but not to DeviceC. After receiving the purge LSP from DeviceC, DeviceE finds that DeviceC does not support source tracing and then generates a PS-PDU which carries information about the advertisement source (DeviceE), purge source (DeviceC), and the purged LSP, and floods the PS-PDU on the network.

After the fault occurs, maintenance personnel can log in to any node on the network except DeviceC to locate the faulty node. Two possible faulty nodes can be located in this case: DeviceA and DeviceC, and they both send the same purge LSP. In this case, DeviceA takes precedence over DeviceC when the maintenance personnel determine the most probable fault source. After DeviceA is isolated, the network recovers, ruling out the possibility that DeviceC is the fault source.

#### Scenario where source tracing-incapable nodes are isolated from source tracing-capable nodes

Assume that all devices except DeviceC and DeviceD support source tracing and DeviceA is the fault source. In this scenario, PS-PDUs cannot be flooded on the entire network. The fault source locating is complicated. [Figure 1-257](#) shows the networking.

**Figure 1-257 Scenario where source tracing-incapable nodes are isolated from source tracing-capable nodes**



When DeviceA purges an IS-IS LSP, it floods a PS-PDU that carries node A information and brief information about the LSP. However, the PS-PDU sent by DeviceA can only reach DeviceB because DeviceC and DeviceD do not support IS-IS purge source tracing.

During source tracing capability negotiation, DeviceE and DeviceF find that DeviceC and DeviceD do not support source tracing, respectively. After receiving the purge LSP from DeviceC, DeviceE generates and floods a PS-PDU on behalf of DeviceC. Similarly, after receiving the purge LSP from DeviceD, DeviceF generates and floods a PS-PDU on behalf of DeviceD.

After the fault occurs, maintenance personnel can locate the fault source (DeviceA) directly if they log in to DeviceA or DeviceB. After DeviceA is isolated, the network recovers. However, if the personnel log in to DeviceE, DeviceF, DeviceG, or DeviceH, they will find that DeviceE claims DeviceC to be the fault source and DeviceF claims DeviceD to be the fault source. If the personnel then log in to DeviceC or DeviceD, they will find that the purge LSP was sent by DeviceB, and was not generated by DeviceC or DeviceD. If the personnel then log in to DeviceB, they will determine that DeviceA is the fault source. After DeviceA is isolated, the network recovers.

## IS-IS MT

With IS-IS multi-topology (MT), IPv6, multicast, and advanced topologies can have their own routing tables. This feature prevents packet loss if an integrated topology and the IPv4/IPv6 dual stack are deployed, isolates multicast services

from unicast routes, improves network resource usage, and reduces network construction cost.

## Context

On a traditional IP network, IPv4 and IPv6 share the same integrated topology, and only one unicast topology exists, which causes the following problems:

- Packet loss if the IPv4/IPv6 dual stack is deployed: If some routers and links in an IPv4/IPv6 topology do not support IPv4 or IPv6, they cannot receive IPv4 or IPv6 packets sent from the router that supports the IPv4/IPv6 dual stack. As a result, these packets are discarded.
- Multicast services highly depending on unicast routes: Only one unicast forwarding table is available on the forwarding plane because only one unicast topology exists, which forces services transmitted from one router to the same destination address to share the same next hop, and various end-to-end services, such as voice and data services, to share the same physical links. As a result, some links may be heavily congested whereas others remain relatively idle. In addition, the multicast reverse path forwarding (RPF) check depends on the unicast routing table. If the default unicast routing table is used when transmitting multicast services, multicast services depend heavily on unicast routes, a multicast distribution tree cannot be planned independently of unicast routes, and unicast route changes affect multicast distribution tree establishment.

Deploying multiple topologies for different services on a physical network can address these problems. IS-IS MT transmits MT information through new TLVs in IS-IS packets. Users can deploy multiple logical topologies based on IP protocols or service types supported by links so that SPF calculations are performed independently in different topologies, which improves network usage.

### NOTE

If an IPv4 or IPv6 BFD session is Down in a topology on a network enabled with MT, neighbors of the IPv4 or IPv6 address family will be affected.

## Related Concepts

IS-IS MT allows multiple route selection subsets to be deployed on a versatile network infrastructure and divides a physical network into multiple logical topologies, where each topology performs its own SPF calculations.

IS-IS MT, an extension of IS-IS, allows multiple topologies to be applied to IS-IS. IS-IS MT complies with standard protocols and transmits multi-topology information using new TLVs in IS-IS packets. Users can deploy multiple logical topologies on a physical network. Each topology performs its own SPF calculations and maintains its own routing table. Traffic of different services, including the traffic transmitted in different IP topologies, has its own optimal forwarding path.

The MT ID configured on an interface identifies the topology bound to the interface. One or more MT IDs can be configured on a single interface.

Reverse path forwarding (RPF) check: After receiving a packet, a device searches its unicast routing table, MBGP routing table, MIGP routing table, and multicast static routing table based on the packet source and selects an optimal route from these routing tables as the RPF route. If the interface that the packet arrives at is

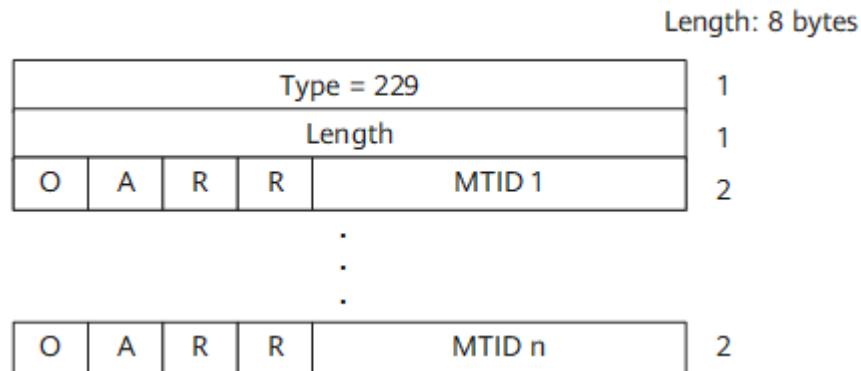
the same as the RPF interface, the packet passes the RPF check and is forwarded. Otherwise, the RPF check fails and traffic is interrupted.

## Implementation

IS-IS MT uses MT IDs to identify different topologies. Each Hello packet or LSP sent by a router carries one or more MT TLVs of the topologies to which the source interface belongs. If the router receives from a neighbor a Hello packet or LSP that carries only some of the local MT TLVs, the router assumes that the neighbor belongs to only the default IPv4 topology. On a point-to-point (P2P) link, an adjacency cannot be established between two neighbors that share no common MT ID. On broadcast links, adjacencies can still be established between neighbors even if they do not share the same MT ID.

[Figure 1-258](#) shows the MT TLV format.

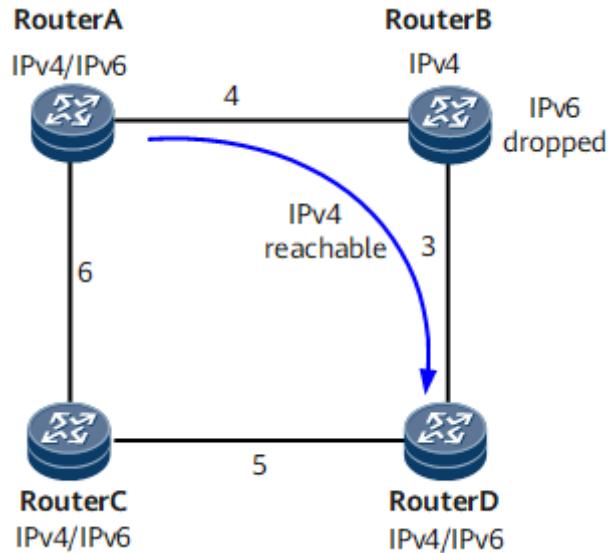
[Figure 1-258](#) MT TLV format



The following section uses IS-IS MT to describe separation of the dual stack and the separation of the multicast topology from the unicast topology.

- [Figure 1-259](#) shows the networking for separation of the IPv4 topology from the IPv6 topology. The values in the networking diagram are link costs. Device A, Device C, and Device D support the IPv4/IPv6 dual stack; Device B supports IPv4 only and cannot forward IPv6 packets.

**Figure 1-259 Separation of the IPv4 topology from the IPv6 topology**

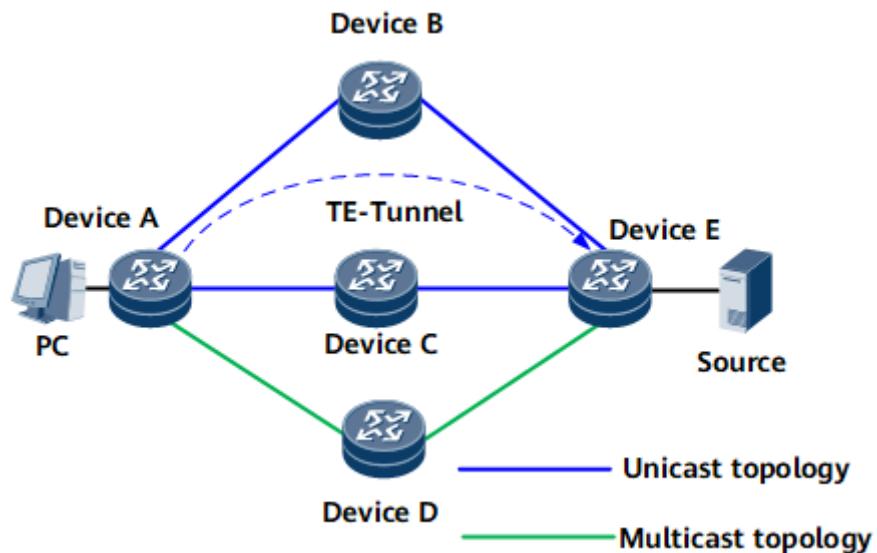


Without IS-IS MT, Device A, Device B, Device C, and Device D use the IPv4/IPv6 topology to perform SPF calculation. In this case, the shortest path from Device A to Device D is Device A -> Device B -> Device D. IPv6 packets cannot reach Device D through Device B because Device B does not support IPv6.

If a separate IPv6 topology is set up using IS-IS MT, Device A chooses only IPv6 links to forward IPv6 packets. In this case, the shortest path from Device A to Device D is Device A -> Device C -> Device D.

- **Figure 1-260** shows the networking for separation between unicast and multicast topologies using IS-IS MT.

**Figure 1-260 Separation of the multicast topology from the unicast topology**



On the network shown in [Figure 1-260](#), all routers are interconnected using IS-IS. A TE tunnel is set up between Device A (ingress) and Device E (egress). The outbound interface of the route calculated by IS-IS may not be a physical interface but a TE tunnel interface. In this case, router C through which the TE tunnel passes cannot set up multicast forwarding entries. As a result, multicast services cannot be transmitted.

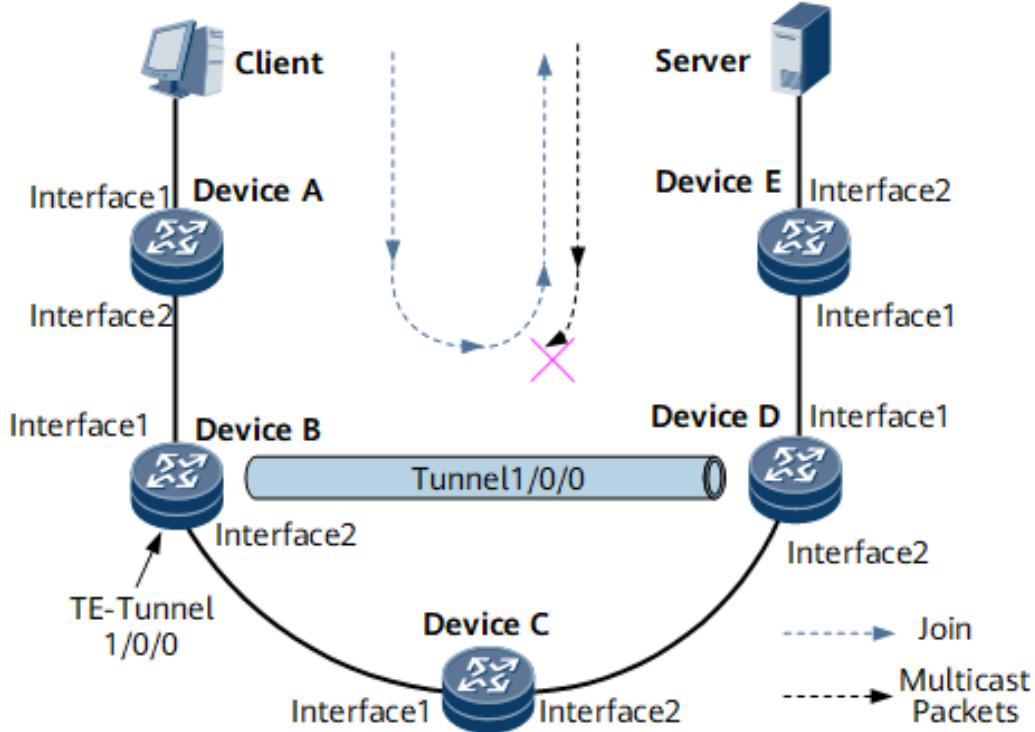
IS-IS MT addresses this problem by establishing separate unicast and multicast topologies. TE tunnels are excluded from a multicast topology. Therefore, multicast services are unaffected by TE tunnels.

## IS-IS Local MT

### Background

On a network where multicast and a unidirectional TE tunnel are deployed, if the TE tunnel is configured with IGP Shortcut, IS-IS uses an MPLS TE tunnel that is up to perform SPF calculation. In this case, the outbound interface of the route calculated by IS-IS may be a TE tunnel interface rather than a physical interface. As a result, the routers spanned by the TE tunnel cannot detect multicast packets and may discard multicast data packets, affecting network reliability. [Figure 1-261](#) shows the networking.

**Figure 1-261** Conflict between multicast and a unidirectional TE tunnel



Client and Server exchange multicast packets as follows:

1. Client sends a Report message to DeviceA, requesting to join a multicast group. Upon receipt, DeviceA sends a Join message to DeviceB.

2. When the Join message reaches DeviceB, DeviceB selects TE-Tunnel1/0/0 as the Reverse Path Forwarding (RPF) interface and forwards the message to DeviceC through Interface 2 based on an MPLS label.
3. Because the Join message is forwarded based on an MPLS label, DeviceC does not create a multicast forwarding entry. As the penultimate hop of the MPLS forwarding, DeviceC removes the MPLS label and forwards the Join message to DeviceD through Interface2.
4. After DeviceD receives the Join message, it generates a multicast forwarding entry in which the upstream and downstream interfaces are Interface1 and Interface2, respectively. DeviceD then sends the Join message to DeviceE. Then the shortest path tree is established.
5. When DeviceD receives traffic from the multicast source, DeviceD sends traffic to DeviceC. Because DeviceC has not created a forwarding entry for the traffic, the traffic is discarded. As a result, multicast services are interrupted.

IS-IS local multicast topology (MT) can address this problem.

## Related Concepts

IS-IS local MT is a mechanism that enables the routing management (RM) module to create a separate multicast topology on the local device so that protocol packets exchanged between devices are not erroneously discarded. When the outbound interface of the route calculated by IS-IS is an IGP Shortcut-enabled TE tunnel interface, IS-IS local MT calculates a physical outbound interface for the route. This mechanism resolves the conflict between multicast and a TE tunnel.

### NOTE

The TE tunnel described in this section is IGP Shortcut-enabled.

## Implementation

[Figure 1-262](#) shows how multicast packets are forwarded after local MT is enabled.

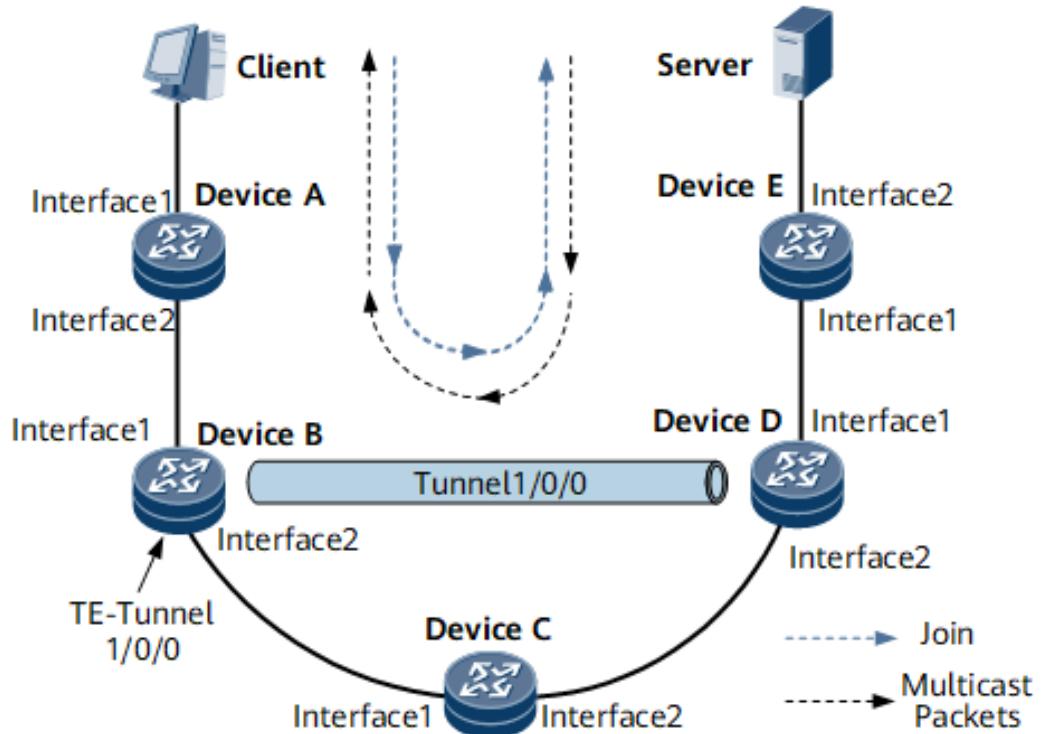
1. Establishment of a multicast IGP (MIGP) routing table

As the Shortcut TE tunnel ingress, DeviceB creates an independent MIGP routing table, records the physical interface corresponding to the TE tunnel interface, and generates multicast routing entries for multicast packet forwarding. If the outbound interface of a calculated route is a TE tunnel interface, IS-IS calculates a physical outbound interface for the route and adds the route to the MIGP routing table.

2. Multicast packet forwarding

When forwarding multicast packets, a router searches the unicast routing table for a route. If the next hop of the route is a tunnel interface, the router searches the MIGP routing table for the physical outbound interface to forward multicast packets. In this example, the original outbound interface of the route is TE tunnel 1/0/0. IS-IS re-calculates a physical outbound interface (Interface2) for the route and adds the route to the MIGP routing table. Multicast services are thus not affected by the TE tunnel. Multicast packets are forwarded through the physical outbound interfaces according to the MIGP routing table. The corresponding routing entries are created in the multicast routing table. Multicast data packets are then correctly forwarded.

Figure 1-262 Local MT networking



## Usage Scenario

IS-IS local MT prevents multicast services from being interrupted on networks, which allows multicasting and has an IGP Shortcut-enabled TE tunnel.

## Benefits

Local MT resolves the conflict between multicast and a TE tunnel and improves multicast service reliability.

## IS-IS Control Messages

IS-IS routers implement routing by exchanging control messages. This section describes IS-IS control messages.

## IS-IS PDU Formats

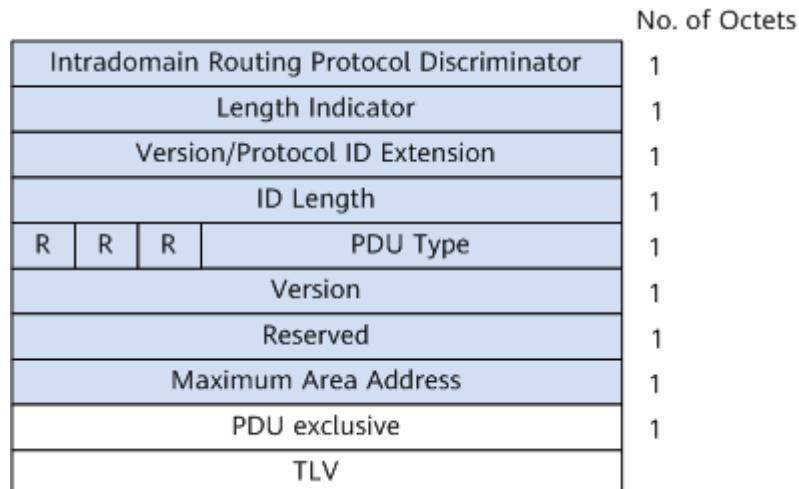
Nine types of IS-IS protocol data units (PDUs) are available for processing control information. Each PDU is identified by a 5-digit type code. IS-IS has three major types of PDUs: Hello PDUs, Link State PDUs (LSPs), and Sequence Number PDUs (SNPs). [Table 1-88](#) shows the mapping between PDUs and type values.

**Table 1-88** Mapping between PDUs and type values

| PDU Type                              | Acronym    | Type Value |
|---------------------------------------|------------|------------|
| Level-1 LAN IS-IS Hello PDU           | L1 LAN IIH | 15         |
| Level-2 LAN IS-IS Hello PDU           | L2 LAN IIH | 16         |
| Point-to-Point IS-IS Hello PDU        | P2P IIH    | 17         |
| Level-1 Link State PDU                | L1 LSP     | 18         |
| Level-2 Link State PDU                | L2 LSP     | 20         |
| Level-1 Complete Sequence Numbers PDU | L1 CSNP    | 24         |
| Level-2 Complete Sequence Numbers PDU | L2 CSNP    | 25         |
| Level-1 Partial Sequence Numbers PDU  | L1 PSNP    | 26         |
| Level-2 Partial Sequence Numbers PDU  | L2 PSNP    | 27         |

The first eight bytes in all IS-IS PDUs are public. [Figure 1-263](#) shows the IS-IS PDU format.

**Figure 1-263** IS-IS PDU format



The main fields are as follows:

- Intradomain Routing Protocol Discriminator: network layer protocol identifier assigned to IS-IS, which is 0x83.
- Length Indicator: length of the fixed header, in bytes.
- ID Length: length of the system ID of network service access point (NSAP) addresses or NETs in this routing domain.
- PDU Type: type of a PDU. For details, see [Table 1-88](#).

- Maximum Area Address: maximum number of area addresses supported by an IS-IS area. The value **0** indicates that a maximum of three area addresses are supported by this IS-IS area.
- Type/Length/Value (TLV): encoding type that features high efficiency and expansibility. Each type of PDU contains a different TLV. **Table 1-89** shows the mapping between TLV codes and PDU types.

**Table 1-89** Mapping between TLV codes and PDU types

| TLV Code | TLV Code Name                             | PDU Type          |
|----------|-------------------------------------------|-------------------|
| 1        | Area Addresses                            | I IH, LSP         |
| 2        | IS Neighbors (LSP)                        | LSP               |
| 4        | Partition Designated Level2 IS            | L2 LSP            |
| 6        | IS Neighbors (MAC Address)                | LAN I IH          |
| 7        | IS Neighbors (SNPA Address)               | LAN I IH          |
| 8        | Padding                                   | I IH              |
| 9        | LSP Entries                               | SNP               |
| 10       | Authentication Information                | I IH, LSP, or SNP |
| 128      | IP Internal Reachability Information      | LSP               |
| 129      | Protocols Supported                       | I IH or LSP       |
| 130      | IP External Reachability Information      | L2 LSP            |
| 131      | Inter-Domain Routing Protocol Information | L2 LSP            |
| 132      | IP Interface Address                      | I IH or LSP       |

## Hello Packet Format

Hello packets, also called the IS-to-IS Hello PDUs (I I H s), are used to set up and maintain neighbor relationships. Level-1 LAN I I H s are applied to the Level-1 routers on broadcast LANs. Level-2 LAN I I H s are applied to the Level-2 routers on broadcast LANs. P2P I I H s are applied to non-broadcast networks. I I H s in different networks have different formats.

- LAN I I H s: **Figure 1-264** shows the format of I I H s on a broadcast network.

**Figure 1-264** Level-1/Level-2 LAN IIH format

|                                             | No. of Octets |
|---------------------------------------------|---------------|
| Intra-domain Routing Protocol Discriminator | 1             |
| Length Indicator                            | 1             |
| Version/Protocol ID Extension               | 1             |
| ID Length                                   | 1             |
| R R R PDU Type                              | 1             |
| Version                                     | 1             |
| Reserved                                    | 1             |
| Maximum Area Address                        | 1             |
| Reserved/Circuit Type                       | 1             |
| Source ID                                   | ID Length     |
| Holding Time                                | 2             |
| PDU Length                                  | 2             |
| R Priority                                  | 1             |
| LAN ID                                      | ID Length+1   |
| Variable Length Fields                      |               |

- P2P IIHs: **Figure 1-265** shows the format of IIHs on a P2P network.

**Figure 1-265** P2P IIH format

|                                             | No. of Octets |
|---------------------------------------------|---------------|
| Intra-domain Routing Protocol Discriminator | 1             |
| Length Indicator                            | 1             |
| Version/Protocol ID Extension               | 1             |
| ID Length                                   | 1             |
| R R R PDU Type                              | 1             |
| Version                                     | 1             |
| Reserved                                    | 1             |
| Maximum Area Address                        | 1             |
| Reserved/Circuit Type                       | 1             |
| Source ID                                   | ID Length     |
| Holding Time                                | 2             |
| PDU Length                                  | 2             |
| Local Circuit ID                            | 1             |
| Variable Length Fields                      |               |

As shown in [Figure 1-265](#), most fields in a P2P IIH are the same as those in a LAN IIH. The P2P IIH does not have the priority and LAN ID fields but has a local circuit ID field. The local circuit ID indicates the local link ID.

## LSP Format

LSPs are used to exchange link-state information. There are two types of LSPs: Level-1 and Level-2. Level-1 IS-IS transmits Level-1 LSPs. Level-2 IS-IS transmits Level-2 LSPs. Level-1-2 IS-IS can transmit both Level-1 and Level-2 LSPs.

Level-1 and Level-2 LSPs have the same format, as shown in [Figure 1-266](#).

**Figure 1-266** Level-1 or Level-2 LSP

|                                            | No. of Octets |
|--------------------------------------------|---------------|
| Intradomain Routing Protocol Discriminator | 1             |
| Length Indicator                           | 1             |
| Version/Protocol ID Extension              | 1             |
| ID Length                                  | 1             |
| R R R PDU Type                             | 1             |
| Version                                    | 1             |
| Reserved                                   | 1             |
| Maximum Area Address                       | 1             |
| PDU Length                                 | 2             |
| Remaining Lifetime                         | 2             |
| LSP ID                                     | ID Length+2   |
| Sequence Number                            | 4             |
| Checksum                                   | 2             |
| R ATT OL IS Type                           | 1             |
| Variable Length Fields                     |               |

The main fields are as follows:

- ATT: Attached bit  
ATT is generated by a Level-1-2 router to identify whether the originating router is connected to other areas. When a Level-1 router receives a Level-1 LSP with ATT as 1 from a Level-1-2 router, the Level-1 router generates a default route destined for the Level-1-2 router so that data can be transmitted to other areas.  
Although ATT is defined in both the Level-1 LSP and Level-2 LSP, it is set only in the Level-1 LSP only by the Level-1-2 router.
- OL: LSDB overload  
LSPs with the overload bit are still flooded on networks, but the LSPs are not used when routes that pass through a device configured with the overload bit are calculated. That is, after a device is configured with the overload bit, other

devices ignore the device when performing the SPF calculation except for the direct routes of the device.

- IS Type: type of the IS-IS generating the LSP

IS Type is used to specify whether the IS-IS type is Level-1 or Level-2 IS-IS. The value **01** indicates Level-1; the value **11** indicates Level-2.

## SNP Format

SNPs describe the LSPs in all or some of the databases and are used to synchronize and maintain all LSDBs. SNPs consist of complete SNPs (CSNPs) and partial SNPs (PSNPs).

- CSNPs carry summaries of all LSPs in LSDBs, which ensures LSDB synchronization between neighboring routers. On a broadcast network, the designated intermediate system (DIS) sends CSNPs at an interval. The default interval is 10 seconds. On a P2P link, neighboring devices send CSNPs only when a neighbor relationship is established for the first time.

[Figure 1-267](#) shows the CSNP format.

[Figure 1-267](#) Level-1/Level-2 CSNP format

|                                             | No. of Octets |
|---------------------------------------------|---------------|
| Intra-domain Routing Protocol Discriminator | 1             |
| Length Indicator                            | 1             |
| Version/Protocol ID Extension               | 1             |
| ID Length                                   | 1             |
| R    R    R    PDU Type                     | 1             |
| Version                                     | 1             |
| Reserved                                    | 1             |
| Maximum Area Address                        | 1             |
| PDU Length                                  | 2             |
| Source ID                                   | ID Length+1   |
| Start LSP ID                                | ID Length+2   |
| End LSP ID                                  | ID Length+2   |
| Variable Length Fields                      |               |

The main fields are as follows:

- Source ID: system ID of the router that sends SNPs
- Start LSP ID: ID of the first LSP in a CSNP
- End LSP ID: ID of the last LSP in a CSNP
- PSNPs list only the sequence numbers of recently received LSPs. A PSNP can acknowledge multiple LSPs at a time. If an LSDB is not updated, PSNPs are also used to request a new LSP from a neighbor.

[Figure 1-268](#) shows the PSNP format.

Figure 1-268 Level-1/Level-2 PSNP format

|                                             |  | No. of Octets |
|---------------------------------------------|--|---------------|
| Intra-domain Routing Protocol Discriminator |  | 1             |
| Length Indicator                            |  | 1             |
| Version/Protocol ID Extension               |  | 1             |
| ID Length                                   |  | 1             |
| R R R PDU Type                              |  | 1             |
| Version                                     |  | 1             |
| Reserved                                    |  | 1             |
| Maximum Area Address                        |  | 1             |
| PDU Length                                  |  | 2             |
| Source ID                                   |  | ID Length+1   |
| Variable Length Fields                      |  |               |

## IS-IS GR

### NOTE

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X can be configured as a GR helper rather than a GR restarter. This function is enabled by default and does not need to be configured additionally. GR is a method of implementing non-stop forwarding (NSF). The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports NSR. NSR has higher function specifications than NSF.

Graceful restart (GR) is a technology that ensures normal data forwarding and prevents key services from being affected when a routing protocol restarts.

When GR is not supported, the active/standby switchover triggered by various reasons causes short-time forwarding interruption and route flapping on the entire network. Route flapping and service interruption are unacceptable on a large-scale network, especially on a carrier network.

GR is an HA technique introduced to resolve the preceding problem. HA technologies comprise a set of comprehensive techniques, such as fault-tolerant redundancy, link protection, faulty node recovery, and traffic engineering. As a fault-tolerant redundancy technology, GR is widely used to ensure non-stop forwarding of key data during the active/standby switchover and system upgrade.

When the GR function is enabled, the forwarding plane continues data forwarding during a restart, and operations on the control plane, such as re-establishment of neighbor relationships and route calculation, do not affect the forwarding plane, preventing service interruption caused by route flapping and improving network reliability.

## Routing Loop Detection for Routes Imported to IS-IS

Routes of an IS-IS process can be imported to another IS-IS process or the process of another protocol (such as OSPF or BGP) for redistribution. However, if a device

that performs such a route import is incorrectly configured, routing loops may occur. IS-IS can use the routing loop detection function to detect routing loops.

## Related Concepts

### Redistribute ID

IS-IS uses a system ID as a redistribution identifier, OSPF and OSPFv3 use a router ID + process ID as a redistribution identifier, and BGP uses a VrfID + random number as a redistribution identifier. For ease of understanding, the redistribution identifiers of different protocols are all called Redistribute IDs. When routes are distributed, the extended TLVs carried in the routes contain Redistribute IDs.

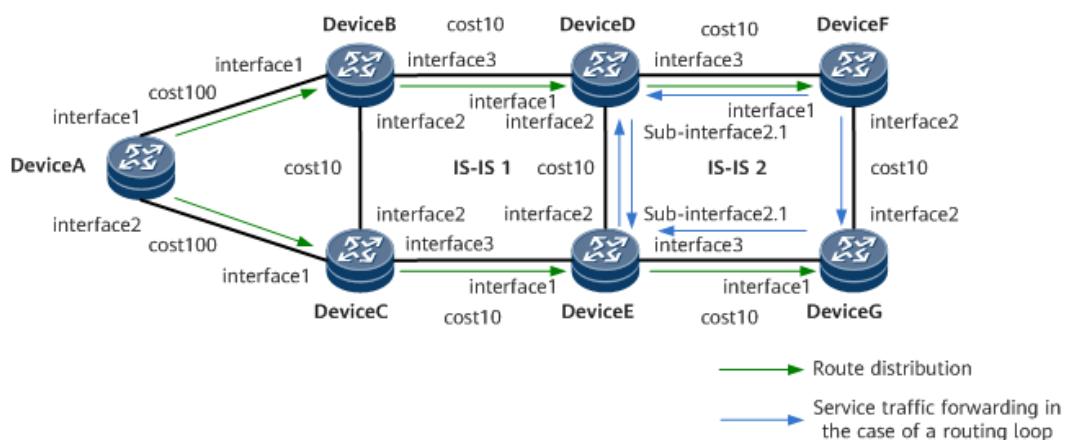
### Redistribute List

A Redistribute list may consist of multiple Redistribute IDs. Each Redistribute list of BGP contains a maximum of four Redistribute IDs, and each Redistribute list of any other routing protocol contains a maximum of two Redistribute IDs. When the number of Redistribute IDs exceeds the corresponding limit, the old ones are discarded according to the sequence in which Redistribute IDs are added.

## Cause (IS-IS Inter-Process Mutual Route Import)

On the network shown in [Figure 1-269](#), DeviceA, DeviceB, and DeviceC run IS-IS process 1, DeviceF and DeviceG run IS-IS process 2, and DeviceD and DeviceE run both processes. DeviceD and DeviceE are configured to import routes between IS-IS processes 1 and 2. The routes distributed by IS-IS process 1 are re-distributed back to IS-IS process 1 through IS-IS process 2. As the costs of the newly distributed routes are smaller, they are preferentially selected, resulting in routing loops.

**Figure 1-269** Typical network diagram of IS-IS inter-process mutual route import



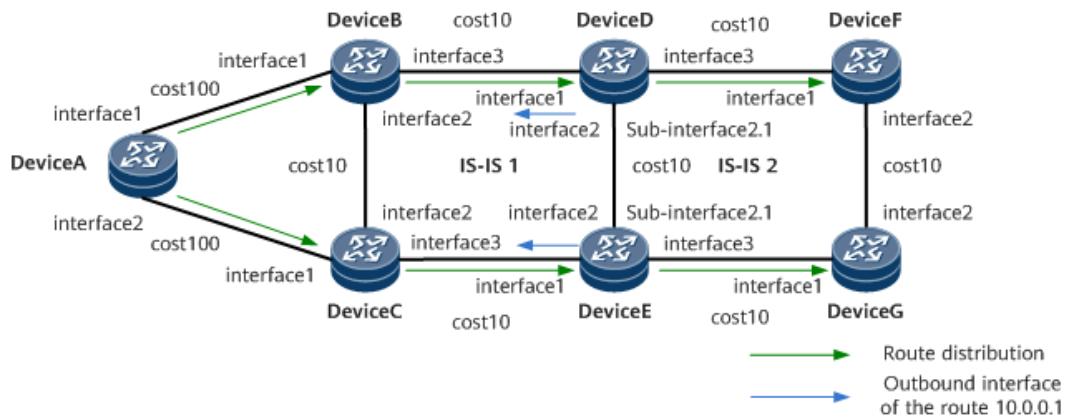
Take DeviceA distributing route 10.0.0.1/32 as an example. A stable routing loop is formed through the following process:

### Phase 1

On the network shown in [Figure 1-270](#), IS-IS process 1 on DeviceA imports the static route 10.0.0.1, generates an LSP carrying the prefix of this route and floods the LSP in IS-IS process 1. After receiving the LSP, IS-IS process 1 on DeviceD and

IS-IS process 1 on DeviceE each calculate a route to 10.0.0.1, with the outbound interface being interface1 on DeviceD and interface1 on DeviceE, respectively, and the cost being 110. At this point, the routes to 10.0.0.1 in IS-IS process 1 in the routing tables of DeviceD and DeviceE are active.

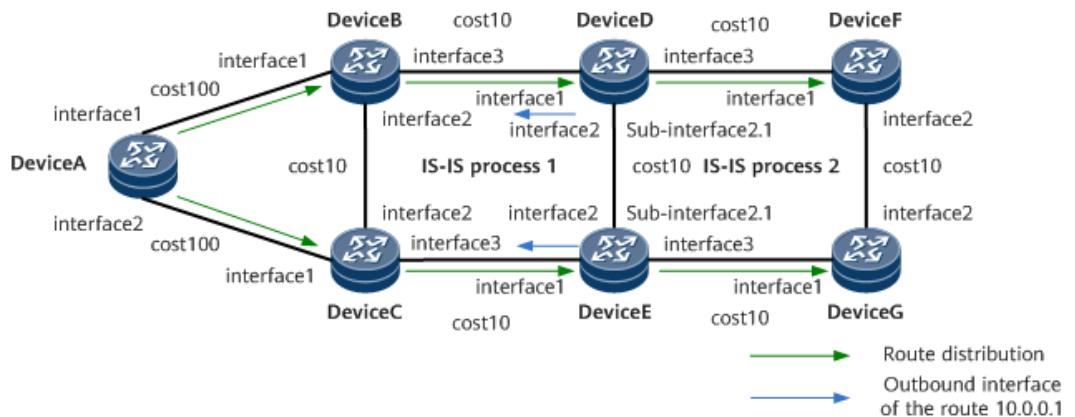
**Figure 1-270 Phase 1**



### Phase 2

In [Figure 1-271](#), DeviceD and DeviceE are configured to import routes from IS-IS process 1 to IS-IS process 2. Either no route-policy is configured for the import or the configured route-policy is improper. DeviceE is used as an example. In phase 1, the route to 10.0.0.1 in IS-IS process 1 in the routing table of DeviceE is active. In this case, IS-IS process 2 imports this route from IS-IS process 1, generates an LSP carrying the prefix of this route, and floods the LSP in IS-IS process 2. After receiving the LSP, IS-IS process 2 on DeviceD calculates a route to 10.0.0.1, with the cost being 10, which is smaller than that (110) of the route calculated by IS-IS process 1. As a result, the active route to 10.0.0.1 in the routing table of DeviceD is switched from the one calculated by IS-IS process 1 to the one calculated by IS-IS process 2, and the outbound interface is sub-interface 2.1.

**Figure 1-271 Phase 2**

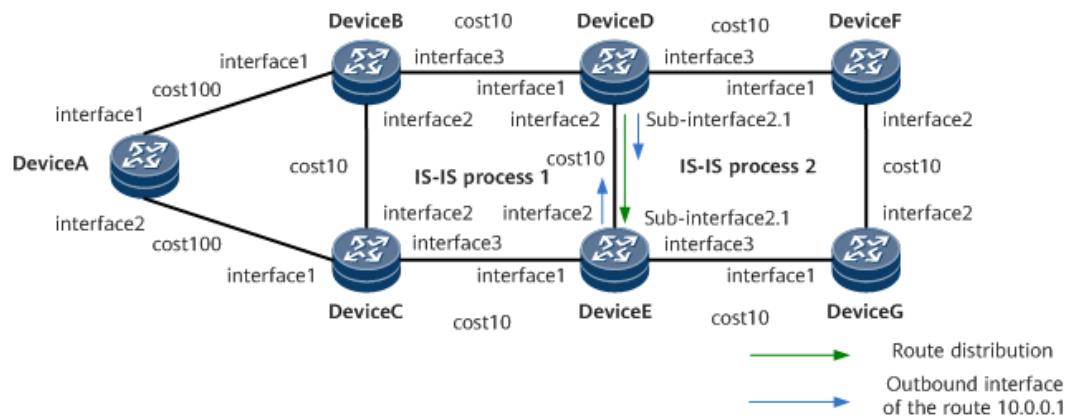


### Phase 3

In [Figure 1-272](#), after the route to 10.0.0.1 in IS-IS process 2 on DeviceD becomes active, IS-IS process 1 imports this route from IS-IS process 2, generates an LSP

carrying the prefix of this route, and floods the LSP in IS-IS process 1. After receiving the LSP, IS-IS process 1 on DeviceE recalculates the route to 10.0.0.1, with the cost being 10, which is smaller than that (110) of the previously calculated route. As a result, the route to 10.0.0.1 in IS-IS process 1 in the routing table of DeviceE is switched to the route (with the smaller cost) advertised by DeviceD, and the outbound interface is interface 2.

**Figure 1-272 Phase 3**

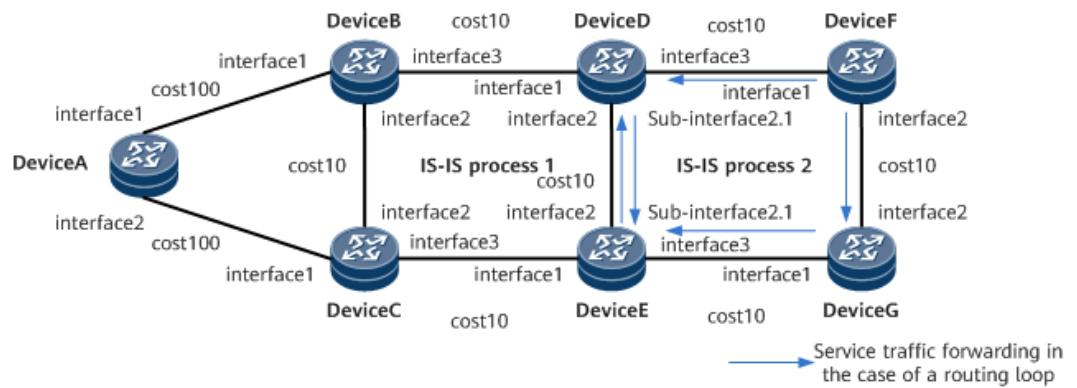


#### Phase 4

After the active route to 10.0.0.1 on DeviceE is updated, IS-IS process 2 still imports the route from IS-IS process 1 as the route remains active, and continues to advertise/update an LSP.

As a result, a stable routing loop is formed. Assuming that traffic is injected from DeviceF, [Figure 1-273](#) shows the traffic flow when the routing loop occurs.

**Figure 1-273 Traffic flow when a routing loop occurs**



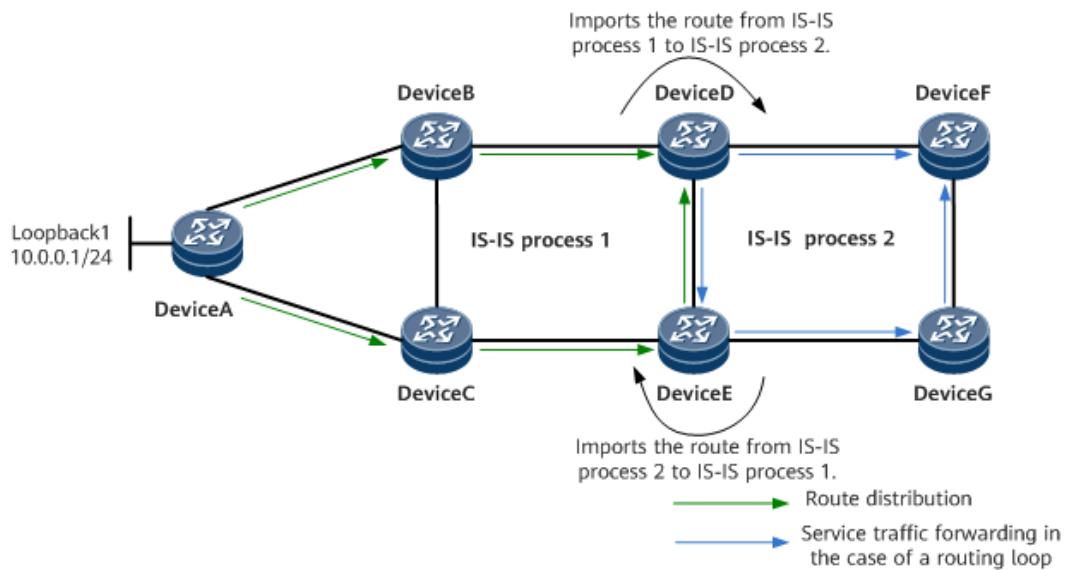
### Implementation (IS-IS Inter-Process Mutual Route Import)

Routing loop detection for IS-IS inter-process mutual route import can resolve the routing loop in the preceding scenario.

When distributing a TLV (with the type value of 135 or 235) for an imported route, IS-IS also uses a sub-TLV (with the type value of 10) of the TLV (with the type value of 135 or 235) to distribute to other devices the Redistribute ID of the device that re-distributes the imported route. If the route is re-distributed by

multiple devices, a maximum of two Redistribute IDs of these devices are distributed through the sub-TLV (with the type value of 10) of the TLV (with the type value of 135 or 235). After receiving the sub-TLV, a route calculation device saves the Redistribute IDs of the re-distribution devices along with the route. When the route is imported by another process, the device checks whether the re-distribution information of the route contains the Redistribute ID of the local process. If the information contains the Redistribute ID of the local process, the device determines that a routing loop occurs and distributes a large route cost in the TLV (with the type value of 135 or 235) for the imported route. This ensures that other devices preferentially select other paths after learning the route, preventing routing loops.

**Figure 1-274** Typical networking of route import to IS-IS



The following uses the networking shown in [Figure 1-274](#) as an example to describe how a routing loop is detected and resolved.

1. DeviceD learns the route distributed by DeviceB through IS-IS process 1 and imports the route from IS-IS process 1 to IS-IS process 2. When distributing the imported route, IS-IS process 2 on DeviceD distributes the Redistribute ID of IS-IS process 2 through the sub-TLV (with the type value of 10) of the TLV (with the type value of 135 or 235). Similarly, IS-IS process 2 on DeviceE learns the route distributed by DeviceD and saves the Redistribute ID distributed by IS-IS process 2 on DeviceD to the routing table during route calculation.
2. When re-distributing the route imported from IS-IS process 2, IS-IS process 1 on DeviceE also distributes the Redistribute ID of IS-IS process 1 on DeviceE through the sub-TLV (with the type value of 10) of the TLV (with the type value of 135 or 235).
3. After learning the route from DeviceE, IS-IS process 1 on DeviceD saves the Redistribute ID distributed by IS-IS process 1 on DeviceE in the routing table during route calculation.
4. When importing the route from IS-IS process 1 to IS-IS process 2, DeviceD finds that the re-distribution information of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an

alarm. IS-IS process 2 on DeviceD distributes a large cost when distributing the imported route so that other devices preferentially select other paths after learning the route. This prevents routing loops.

 NOTE

In the preceding typical networking:

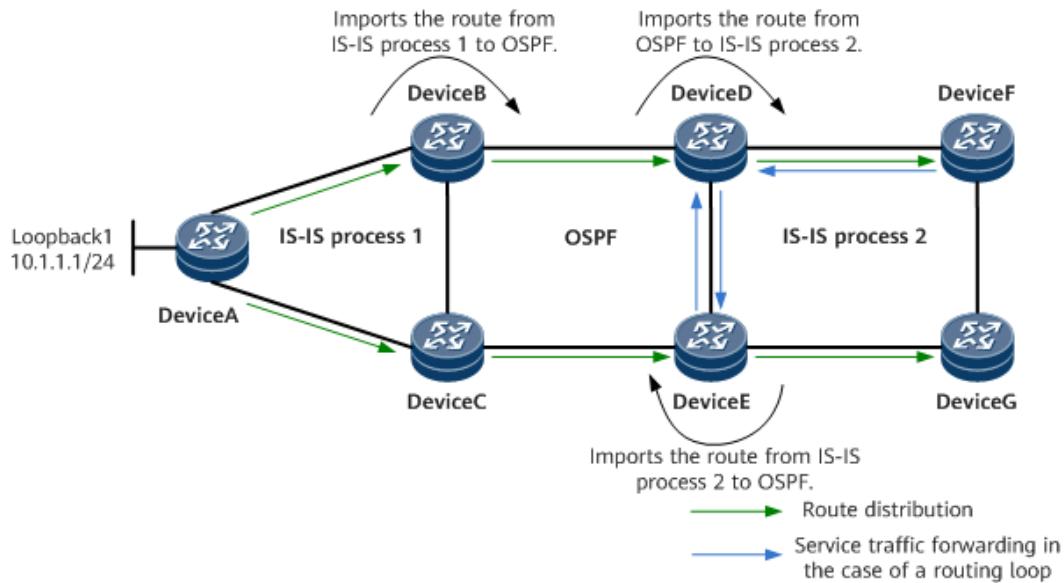
If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

## Cause (Mutual Route Import Between IS-IS and OSPF)

In [Figure 1-275](#), DeviceA, DeviceB, and DeviceC run IS-IS process 1; DeviceD, DeviceE, DeviceF, and DeviceG run IS-IS process 2; in addition, DeviceB, DeviceC, DeviceD, and DeviceE run an OSPF process. DeviceB imports routes of IS-IS process 1 to OSPF, DeviceD imports OSPF routes to IS-IS process 2, and DeviceE imports routes of IS-IS process 2 to OSPF. Improper route import configurations may cause routing loops. For example, if DeviceD preferentially selects routes learned from DeviceE, a routing loop occurs between DeviceD and DeviceE. The following part describes how the routing loop is detected and resolved. Routing loop detection for routes imported to IS-IS and routing loop detection for routes imported to OSPF are enabled by default and do not need to be manually configured.

**Figure 1-275** Typical networking of route import from OSPF to IS-IS



## Implementation (Mutual Route Import Between IS-IS and OSPF)

The following uses the networking shown in [Figure 1-275](#) as an example to describe how a routing loop is detected and resolved.

1. DeviceA distributes its locally originated route 10.1.1.1/24 to DeviceB through IS-IS process 1. DeviceB imports the route from IS-IS process 1 to OSPF and adds the Redistribute ID of OSPF on DeviceB to the route when distributing the route through OSPF.
2. After learning the Redistribute list carried in the route advertised by DeviceB, OSPF on DeviceD saves the Redistribute ID of OSPF on DeviceB to the routing table during route calculation. After DeviceD imports this route from OSPF to IS-IS process 2, DeviceD redistributes the route through IS-IS process 2. In the redistributed route, the extended TLV contains the Redistribute ID of IS-IS process 2 on DeviceD and the Redistribute ID of OSPF on DeviceB. After learning the Redistribute list carried in the route advertised by DeviceD, IS-IS process 2 on DeviceE saves the Redistribute list in the routing table during route calculation.
3. After DeviceE imports this route from IS-IS process 2 to OSPF, DeviceE redistributes the route through OSPF. The redistributed route carries the Redistribute ID of OSPF on DeviceE and the Redistribute ID of IS-IS process 2 on DeviceD. The Redistribute ID of OSPF on DeviceB has been discarded from the route. DeviceD learns the Redistribute list carried in the route distributed by DeviceE and saves the Redistribute list in the routing table. When importing the OSPF route to IS-IS process 2, DeviceD finds that the Redistribute list of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. To resolve the routing loop, IS-IS process 2 on DeviceD distributes a large route cost when redistributing the route. However, because IS-IS has a higher preference than OSPF ASE, DeviceE still prefers the route learned from DeviceD through IS-IS process 2. As a result, the routing loop is not eliminated. The route received by DeviceE carries the Redistribute ID of OSPF on DeviceE and the Redistribute ID of IS-IS process 2 on DeviceD.
4. When importing the route from IS-IS process 2 to OSPF, DeviceE finds that the Redistribution information of the route contains its own Redistribute ID, considers that a routing loop is detected, and reports an alarm. To resolve the routing loop, OSPF on DeviceE distributes a large route cost when redistributing the route. In this case, DeviceD prefers the route distributed by DeviceB. As such, the routing loop is resolved.

 **NOTE**

In the preceding typical networking:

If routes are imported within a protocol on a device and the device detects a routing loop, it increases the cost of the route to be advertised. After the remote device learns this route with a large cost, it does not preferentially select this route as the optimal route in the IP routing table. In this manner, the routing loop is eliminated.

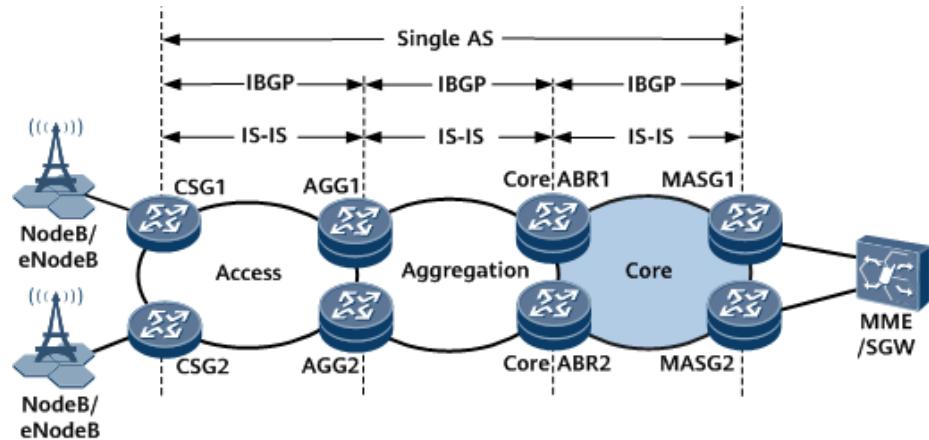
In the case of inter-protocol route import, if a routing protocol with a higher priority detects a routing loop, although this protocol increases the cost of the corresponding route, the cost increase will not render the route inactive. As a result, the routing loop cannot be eliminated. If the routing protocol with a lower priority detects a routing loop and increases the cost of the corresponding route, the originally imported route is preferentially selected. In this case, the routing loop can be eliminated.

## Application Scenarios

**Figure 1-276** shows a typical intra-AS seamless MPLS network. If the IS-IS process deployed at the access layer differs from that deployed at the aggregation layer, IS-IS inter-process mutual route import is usually configured on AGGs so that

routes can be leaked between the access and aggregation layers. As a result, a routing loop may occur between AGG1 and AGG2. If routing loop detection for IS-IS inter-process mutual route import is configured on AGG1 and AGG2, routing loops can be quickly detected and other routes can be preferentially selected, preventing routing loops.

**Figure 1-276** Routing protocol deployment for the intra-AS seamless MPLS networking

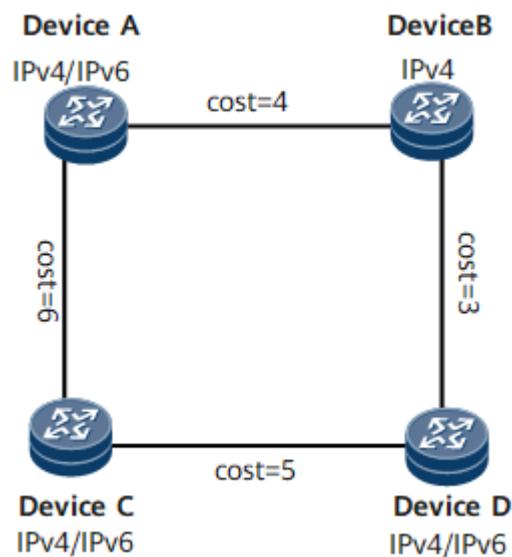


### 1.1.8.1.3 Application Scenarios for IS-IS

#### IS-IS MT

**Figure 1-277** shows the use of IS-IS MT to separate an IPv4 topology from an IPv6 topology. Device A, Device C, and Device D support IPv4/IPv6 dual-stack; Device B supports IPv4 only and cannot forward IPv6 packets.

**Figure 1-277** Separation of the IPv4 topology from the IPv6 topology

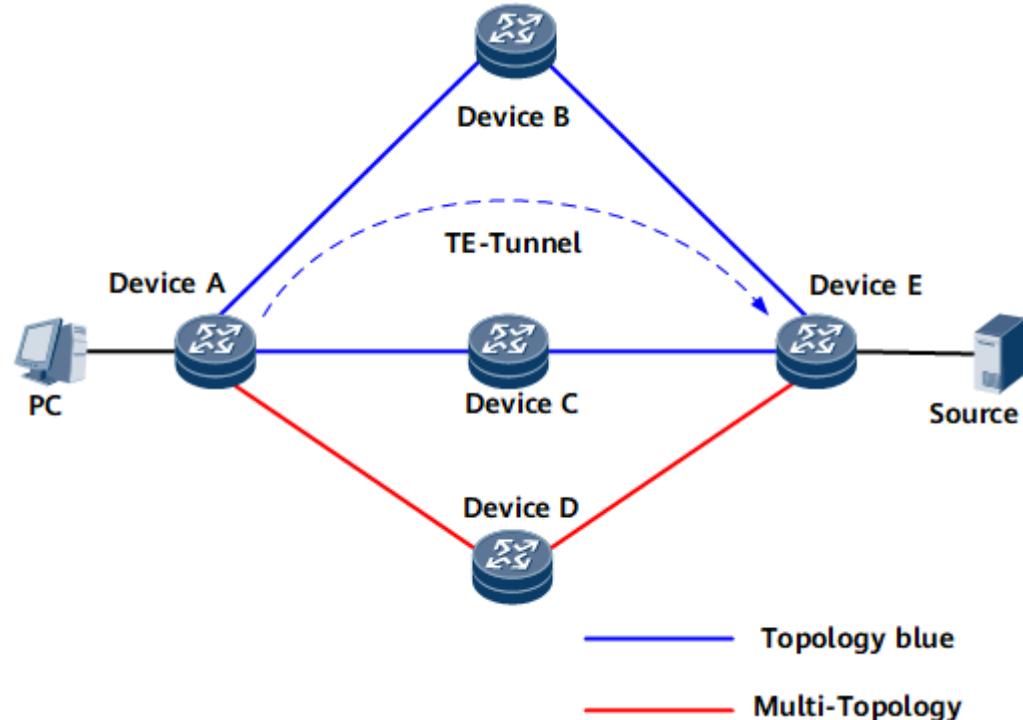


If IS-IS MT is not used, Device A, Device B, Device C, and Device D consider the IPv4 and IPv6 topologies the same when using the SPF algorithm for route calculation. The shortest path from Device A to Device D is Device A -> Device B -> Device D. Device B does not support IPv6 and cannot forward IPv6 packets to Device D.

If IS-IS MT is used to establish a separate IPv6 topology, Device A chooses only IPv6 links to forward IPv6 packets. The shortest path from Device A to Device D changes to Device A -> Device C -> Device D. IPv6 packets are then forwarded.

**Figure 1-278** shows the use of IS-IS MT to separate unicast and multicast topologies.

**Figure 1-278 Separation of the multicast topology from the unicast topology**



All routers in **Figure 1-278** are interconnected using IS-IS. A TE tunnel is set up between Device A (ingress) and Device E (egress). The outbound interface of the route calculated by IS-IS may not be a physical interface but a TE tunnel interface. The routers between which the TE tunnel is established cannot set up multicast forwarding entries. As a result, multicast services cannot run properly.

IS-IS MT is configured to solve this problem by establishing separate unicast and multicast topologies. TE tunnels are excluded from a multicast topology; therefore, multicast services can run properly, without being affected by TE tunnels.

### 1.1.8.2 IS-IS Configuration

This chapter describes the principles and provides configuration procedures and configuration examples of Intermediate System to Intermediate System (IS-IS).

### 1.1.8.2.1 Overview of IS-IS

IS-IS can be used to implement interworking on large-scale networks.

#### Definition

Intermediate System to Intermediate System (IS-IS) is a dynamic routing protocol initially designed by the International Organization for Standardization (ISO) for the Connectionless Network Protocol (CLNP).

To support IP routing, the IETF extends and modifies IS-IS in relevant standards, which enables IS-IS to be applied to both TCP/IP and Open System Interconnection (OSI) environments. The new type of IS-IS is called integrated IS-IS or dual IS-IS.

In this document, IS-IS refers to integrated IS-IS, unless otherwise stated.

#### NOTE

Unless otherwise specified, IS-IS features that support both IPv4 and IPv6 are implemented in the same way.

#### Purpose

IS-IS is an Interior Gateway Protocol (IGP) and is used within an autonomous system (AS). IS-IS is a link state protocol, and it uses the shortest path first (SPF) algorithm to calculate routes.

### 1.1.8.2.2 Configuration Precautions for IS-IS

#### Feature Requirements

**Table 1-90** Feature requirements

| Feature Requirements                                                                                                                                                                                                                                                                                                          | Series           | Models                                                                                     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| In IS-IS broadcast, P2P two-way handshake, IS-IS multi-instance, and IS-IS multi-topology networking scenarios, BFD session establishment check does not take effect even if the bfd session-up check command is run.<br>Association between BFD and IS-IS neighbor Up does not support broadcast interfaces or advanced MTs. | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| After SR LSP strict check is configured, the SR capability of a node is strictly checked when LSPs are calculated from the current node to all nodes. If SR-LDP interworking has been deployed on the network, IS-IS SR-LDP interworking may fail, causing traffic interruptions.                                             | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

| Feature Requirements                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | Series           | Models                                                                                     |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| <p>When an IS-IS Level-12 router imports a route with a priority lower than that of an IS-IS router, the route cannot be learned by another Level of the current router after the route is leaked.</p> <p>When an IS-IS Level-12 router imports a route with a priority lower than that of an IS-IS route, ensure that the route is not learned by another level of the current device after the route is leaked. Otherwise, route flapping occurs. Currently, this problem can be avoided only through configuration.</p> | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>When services without private network labels enter a Policy, the last SID of the Policy must carry the USD attribute, and this SID must be the SID of the egress. Otherwise, traffic cannot be forwarded.</p>                                                                                                                                                                                                                                                                                                           | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>In Encap mode, when the last SID of a Policy is a BSID, traffic fails to be forwarded.</p>                                                                                                                                                                                                                                                                                                                                                                                                                              | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>When an IS-IS Level-1-2 router imports a route with a lower priority than IS-IS, the route cannot be learned by another level of the current router after being leaked. Otherwise, route flapping occurs.</p> <p>You are advised to plan the network properly.</p>                                                                                                                                                                                                                                                      | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>When multiple import points exist and different devices are configured with the same system ID, a false loop alarm may be reported.</p>                                                                                                                                                                                                                                                                                                                                                                                 | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

| Feature Requirements                                                                                                                                                                                                                                                                                                                                                                                                                                                               | Series           | Models                                                                                     |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| <p>When load balancing is used:</p> <ol style="list-style-type: none"> <li>1. TI-LFA backup next hop calculation is not supported in load balancing scenarios.</li> <li>2. FRR is not supported in load balancing scenarios where multiple nodes advertise the same route.</li> <li>3. FRR calculation is not supported in SR-MPLS BE load balancing scenarios.</li> </ol> <p>Therefore, plan the network properly.</p>                                                            | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <ol style="list-style-type: none"> <li>1. The mapping SID does not support priority comparison.</li> <li>2. The mapping SID cannot be used in anycast FRR.</li> <li>3. The mapping SID cannot be used in the SR anti-micro-loop function.</li> <li>4. in cross-process and cross-protocol scenarios, when ASBRs function as interworking stitching nodes, mapping-SID mapping needs to be configured on the processes to be imported.</li> </ol> <p>Properly plan the network.</p> | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>In SR and LDP interworking scenarios, only the mapping SID with the 32-bit mask IP prefix is supported. The SR interworking function does not take effect for the mapping SID mapped to an IP prefix with a non-32-bit mask.</p> <p>Properly plan the network.</p>                                                                                                                                                                                                              | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>In an SR-LDP interworking scenario, FRR does not take effect in the following scenarios:</p> <p>In the SR-to-LDP direction, if there are NEs that do not support SR on the Q node and the path before the Q node,</p> <p>In the LDP-to-SR direction, some NEs on the PQ and previous paths do not support LDP.</p> <p>Plan the network properly.</p>                                                                                                                            | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| <p>SBFD is not supported in SR and LDP interworking scenarios because LDP does not support SBFD.</p> <p>Properly plan the network.</p>                                                                                                                                                                                                                                                                                                                                             | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

| Feature Requirements                                                                                                                                                                                                                                                                                                                            | Series           | Models                                                                                     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| After the device is restarted, if the BFD session of the local device or its neighbor is in Admin Down state, the IS-IS status is not affected. When the BFD session is renegotiated, if the BFD detection status reported by BFD is Down (used to be Up), the IS-IS neighbor is set to Down. In other cases, the IS-IS status is not affected. | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| The association between BFD and IS-IS neighbor Up does not take effect on broadcast interfaces or advanced MT.                                                                                                                                                                                                                                  | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

### 1.1.8.2.3 Configuring Basic IPv4 IS-IS Functions

This section describes how to configure basic IPv4 IS-IS functions.

#### Usage Scenario

To deploy IS-IS on an IPv4 network, configure basic IS-IS functions for communication between different nodes on the network.

Other IS-IS functions can be configured only after basic IS-IS functions are configured.

Configuring basic IPv4 IS-IS functions includes the following operations:

1. Create IPv4 IS-IS processes.
2. Configure IPv4 IS-IS interfaces.

#### Pre-configuration Tasks

Before configuring basic IPv4 IS-IS functions, complete the following tasks:

- Configure a link layer protocol.
- Assign an IP address to each interface to ensure IP connectivity.

#### Creating an IPv4 IS-IS Process

To configure basic IPv4 IS-IS functions, first create an IPv4 IS-IS process and enable IPv4 IS-IS interfaces.

#### Context

To create an IS-IS process, perform the following operations:

- **Create an IS-IS process and configure the NET of a device.**

- **(Optional) Configure the level of a device.**  
Configure the device level based on the network planning. If no device level is configured, IS-IS establishes separate neighbor relationships for Level-1 and Level-2 devices and maintains two identical LSDBs, consuming excessive system resources.
- **(Optional) Configure IS-IS host name mapping.**  
After IS-IS host name mapping is configured, a host name rather than the system ID of a device is displayed when you run a display command. This configuration improves the maintainability on an IS-IS network.
- **(Optional) Enable IS-IS adjacency strict-check.**  
If both IPv4 and IPv6 run on a network, and the IPv6 topology type of this network is **standard** or **compatible**, enable IS-IS adjacency strict-check to ensure that an IS-IS adjacency is established only when both IPv4 and IPv6 go Up. IS-IS adjacency strict-check improves network reliability and prevents traffic losses.
- **(Optional) Enable the LSDB capacity threshold alarm function.**  
An IS-IS-capable router may advertise a large number of LSPs due to an excessive number of external routes received because of incorrect configurations or attack packets. In this situation, you can configure alarm and clear alarm thresholds for the number of LSPs in the LSDB. When the proportion of the number of LSPs to the maximum number of LSPs in the LSDB exceeds the alarm threshold, an alarm is generated; when the proportion of the number of LSPs to the maximum number of LSPs in the LSDB falls below the clear alarm threshold, a clear alarm is generated.
- **(Optional) Configure IS-IS to add purge originator identification (POI) TLV to purge LSPs.**  
This function helps locate the source of error packets when a fault occurs on the network.
- **(Optional) Disable automatic IS-IS system ID recovery in case of conflicts.**  
Two devices on an IS-IS network cannot have the same system ID. Otherwise, network flapping may occur.

## Procedure

- Create an IS-IS process and configure the NET of a device.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
An IS-IS process is created, and its view is displayed.  
*process-id* specifies an IS-IS process ID. If the *process-id* parameter is not specified, the system creates process 1 by default. To associate an IS-IS process with a VPN instance, run the **isis process-id vpn-instance vpn-instance-name** command.
  - c. Run **network-entity net-addr**  
A NET is configured.  
NET of IS-IS consists of three parts:

- Area ID, which is variable (1 to 13 bytes).
- System ID (6 bytes) of this device.
- SEL, which is 1 byte and its value must be 00.

For example, the NET of IS-IS device can be configured as 10.1234.6e9f.0001.00.

#### NOTICE

- An area ID is used to uniquely identify an area in the same IS-IS domain. All routers in the same Level-1 area must share the same area ID, while routers in the same Level-2 area can have different area IDs.
- The system ID must be unique in the whole area and backbone area.
- Multiple NETs can be configured, but they must have the same system ID.

In conclusion, it is recommended that you convert the loopback interface address into a NET to ensure that the NET is unique on the network. If NETs are not unique on the network, route flapping may occur. Therefore, you need to plan the network properly. To convert the IP address of a loopback interface into a system ID, extend each decimal number of the IP address to 3 bits by adding 0s to the front of each decimal number that is shorter than 3 bits, and divide the extended IP address into three parts, with each part consisting of four digits.

d. (Optional) Run **description** *description*

A description is configured for the IS-IS process.

Configuring a description for an IS-IS process facilitates device maintenance and management. The description is not advertised through LSPs.

e. Run **commit**

The configuration is committed.

- (Optional) Configure the level of a device.
  - a. Run **is-level { level-1 | level-1-2 | level-2 }**

The level of the router is configured.
  - b. Run **commit**

The configuration is committed.
- (Optional) Configure IS-IS host name mapping.
  - a. Run **is-name** *symbolic-name*

IS-IS dynamic host name mapping is configured. The system ID of the local device is mapped to the specified host name.

This configuration is dynamic. That is, the configured *symbolic-name* is advertised to other IS-IS devices in the area through LSPs.

The configured host name (*symbolic-name*), rather than the system ID of the local device, is displayed on the IS-IS neighbors when IS-IS information is checked on these neighbors.

- b. Run **is-name map system-id symbolic-name**

IS-IS static host name mapping is configured. The system ID of a peer IS-IS device is mapped to the specified host name.

This command configuration takes effect only on the local IS-IS device. The value of *symbolic-name* will not be added to LSPs.

If dynamic host name mappings are configured on an IS-IS network, the mappings on the network override those statically configured on the local router.

- c. Run **commit**

The configuration is committed.

- (Optional) Enable IS-IS adjacency strict-check.

- a. Run **adjacency-strict-check enable**

IS-IS adjacency strict-check is enabled.

- b. Run **commit**

The configuration is committed.

- (Optional) Enable the LSDB capacity threshold alarm function.

- a. Run **lsdb limit limit-number threshold-alarm upper-limit-value lower-limit lower-limit-value**

The alarm function is enabled for the number of LSPs in the LSDB.

#### NOTE

*upper-limit-value* must be greater than or equal to *lower-limit-value*.

- b. Run **commit**

The configuration is committed.

- (Optional) Configure IS-IS to add POI TLV to purge LSPs.

- a. Run the **purge-originator-identification enable [ always ]** command to configure IS-IS to add the POI TLV (as well as the hostname TLV if the dynamic hostname function is enabled) to purge LSPs.

- If the **purge-originator-identification enable** command is run and the **send-only** parameter is specified when configuring authentication, generated purge LSPs do not carry the POI TLV or hostname TLV.

- If the **purge-originator-identification enable** command is run and HMAC-MD5 authentication is configured, generated purge LSPs do not carry the POI TLV or hostname TLV. If the command is run and authentication of another type is configured or no authentication is configured, generated purge LSPs carry the POI TLV and hostname TLV.

 NOTE

HMAC-SHA256 rather than HMAC-MD5 is recommended for the sake of security.

- If the **purge-originator-identification enable always** command is run, generated purge LSPs carry the POI TLV and hostname TLV, regardless of whether authentication is configured or whether the **send-only** parameter is specified when configuring authentication.

b. Run **commit**

The configuration is committed.

- (Optional) Disable automatic IS-IS system ID recovery in case of conflicts.

a. Run **quit**

Return to the system view.

b. Run **isis system-id auto-recover disable**

Automatic IS-IS system ID recovery is disabled in case of conflicts.

c. Run **commit**

The configuration is committed.

----End

## Configuring an IPv4 IS-IS Interface

To configure an interface on an IS-IS device to send Hello packets or flood LSPs, IS-IS must be enabled on this interface.

### Context

The level of an IS-IS device and level of an interface determine the level of a neighbor relationship. By default, Level-1 and Level-2 neighbor relationships will be established between two Level-1-2 devices. If only one level of neighbor relationships is required, you can configure the level of an interface to prevent the establishment of neighbor relationships of the other level.

After IS-IS is enabled on an interface, the interface will automatically send Hello packets, attempting to establish neighbor relationships. If a peer device is not an IS-IS device or if an interface is not expected to send Hello packets, suppress the interface. Then this interface only advertises routes of the network segment where the interface resides but does not send Hello packets. This suppression reduces the link bandwidth usage.

### Procedure

- Configure an IS-IS interface.
  - a. Run **system-view**

The system view is displayed.
  - b. (Optional) Run **isis interface limit disable**

The limit on the number of IS-IS interfaces that can be configured on the device is disabled.

After this command is run, the number of IS-IS interfaces is no longer limited to the device threshold. To restore the limit on the device, run the **undo isis interface limit disable** command. If the number of IS-IS interfaces on the device is greater than or equal to the threshold before the configuration, no more IS-IS interfaces can be added on the device.

- c. Run **interface interface-type interface-number**

The interface view is displayed.

- d. Run **isis enable [ process-id ]**

An IS-IS interface is configured.

After this command is run, the IS-IS device uses the specified interface to send Hello packets and flood LSPs.

 NOTE

No neighbor relationship needs to be established between loopback interfaces. Therefore, if this command is run on a loopback interface, the routes of the network segment where the loopback interface resides will be advertised through other IS-IS interfaces.

- e. Run **commit**

The configuration is committed.

- (Optional) Configure the level of an IS-IS interface.

- a. Run **isis circuit-level [ level-1 | level-1-2 | level-2 ]**

The level of the interface is configured.

 NOTE

Changing the level of an IS-IS interface is valid only when the level of the IS-IS device is Level-1-2. If the level of the IS-IS device is not a Level-1-2, the level of the IS-IS device determines the level of the adjacency to be established.

- b. Run **commit**

The configuration is committed.

- (Optional) Suppress an IS-IS interface.

- a. Run **isis silent [ advertise-zero-cost ]**

The IS-IS interface is suppressed.

A suppressed IS-IS interface does not send or receive IS-IS packets. The routes of the network segment where the interface resides, however, can still be advertised to other devices within the area.

- b. Run **commit**

The configuration is committed.

----End

## (Optional) Configuring Costs for IPv4 IS-IS Interfaces

Configuring the IS-IS interface costs can control IS-IS route selection.

## Context

The costs of IS-IS interfaces can be determined in the following modes (in descending order of priority):

- The interface cost takes effect only on a specified interface.
- The global cost takes effect only on all interfaces.
- The automatically calculated cost is a cost automatically calculated based on the interface bandwidth.

The default cost of an IS-IS interface is 10, and the default cost style is narrow.

## Procedure

- Configure an IS-IS cost style.

- a. Run **system-view**

The system view is displayed.

- b. Run **isis [ process-id ]**

The IS-IS view is displayed.

- c. Run **cost-style { narrow | wide | wide-compatible | { compatible | narrow-compatible } [ relax-spf-limit ] }**

An IS-IS cost style is configured.

The cost range of an interface and the cost range of routes that the interface accepts vary with the cost style.

- If the cost style is narrow, the cost of an interface ranges from 1 to 63. The maximum cost of routes that the interface accepts is 1023.
- If the cost style is narrow-compatible or compatible, the cost of an interface ranges from 1 to 63. The cost of routes that the interface accepts is related to the parameter **relax-spf-limit**.
  - If **relax-spf-limit** is not specified, the route cost is determined by the following rules:
    - If the cost of the route is not greater than 1023 and the link cost of every interface through which the route passes is less than or equal to 63, the local device accepts the actual cost of the route.
    - If the cost of the route is not greater than 1023, but the link cost of an interface through which the route passes is greater than 63, the device can learn only the routes to the network segment where the interface resides and the routes imported by the interface. The local device considers the cost of the route as the actual one and accepts the route but discards subsequent routes forwarded through the interface.
    - If the cost of the route is greater than 1023, the device can learn only the routes of the interface on which the link cost of the route exceeds 1023 for the first time. The link cost of each interface through which the route passes before the route reaches this interface is not greater than 63. The local device can learn routes to the network segment where the interface resides

- and the routes imported by the interface. The local device considers the cost of the routes as 1023 and accept them but discards subsequent routes forwarded through the interface.
- If **relax-spf-limit** is specified, the route cost is determined by the following rules:

There is no limit on link costs of interfaces or route costs. The local device considers the cost of each route as the actual one and accepts the routes.

    - If the cost style is wide-compatible or wide, the cost of the interface ranges from 1 to 16777214 or **maximum** (16777215). When the cost is **maximum**, the neighbor TLV with cost 16777215 generated on the link is used not for route calculation but for the transmission of TE information. The maximum cost of accepted routes is 0xFFFFFFFF.
- d. Run **commit**
- The configuration is committed.
- Configure a cost for the IS-IS interface.
    - a. Run **system-view**
    - b. Run **interface interface-type interface-number**
    - c. Run **isis [ topology topology-name ] cost cost [ level-1 | level-2 ]**
- A cost is configured for the IS-IS interface.
- The cost set using this command takes effect only on this interface.
- d. Run **commit**
- The configuration is committed.
- Configure the global IS-IS cost.
    - a. Run **system-view**
    - b. Run **isis [ process-id ]**
    - c. Run **circuit-cost { cost } [ level-1 | level-2 ]**
- The global IS-IS cost is configured.
- The cost set using this command takes effect on all interfaces.
- d. Run **commit**
- The configuration is committed.
- Enable IS-IS to automatically calculate interface costs.
    - a. Run **system-view**
    - b. Run **isis [ process-id ]**
- The IS-IS view is displayed.

c. Run **bandwidth-reference value**

The reference value of the bandwidth is configured.

d. Run **auto-cost enable [ compatible ]**

IS-IS is enabled to automatically calculate the interface cost.

- If the cost style of the system is wide or wide-compatible, when **auto-cost enable** command is configured, Interface cost = (Bandwidth-reference/Interface bandwidth) x 10, and when **auto-cost enable compatible** command is configured, Interface cost = (Bandwidth-reference/Interface bandwidth).

 NOTE

If the interface cost calculated through this formula is greater than 16777214, 16777214 is used as the interface cost for route calculation. That is, the interface cost will never be greater than 16777214.

The **auto-cost enable** command can be run on Eth-Trunk interfaces as same with on physical interfaces. If the command is run on an Eth-Trunk interface, the bandwidth of the Eth-Trunk interface is equal to the total bandwidth of all its member interfaces.

- If the cost-style is narrow, narrow-compatible, or compatible, the cost of each interface is determined by the interface bandwidth range. **Table 1-91** lists the interface costs corresponding to different interface bandwidth ranges.

**Table 1-91** Mapping between IS-IS interface costs and interface bandwidth

| Cost | Bandwidth Range                                    |
|------|----------------------------------------------------|
| 60   | Interface bandwidth $\leq$ 10 Mbit/s               |
| 50   | 10 Mbit/s < interface bandwidth $\leq$ 100 Mbit/s  |
| 40   | 100 Mbit/s < interface bandwidth $\leq$ 155 Mbit/s |
| 30   | 155 Mbit/s < interface bandwidth $\leq$ 622 Mbit/s |
| 20   | 622 Mbit/s < Interface bandwidth $\leq$ 2.5 Gbit/s |
| 10   | Interface bandwidth $>$ 2.5 Gbit/s                 |

 NOTE

To change the cost of a loopback interface, run the **isis cost** command only in the loopback interface view.

e. Run **commit**

The configuration is committed.

- Associate the remaining bandwidth of an IS-IS interface with the link cost.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface interface-type interface-number**  
The interface view is displayed.
  - c. Run **isis [ process-id process-id-value ] cost cost-value { higher-bandwidth higher-bandwidth-value cost better-cost-value | lower-bandwidth lower-bandwidth-value cost worse-cost-value } \* [ level-1 | level-2 ]**  
The remaining bandwidth of the IS-IS interface is associated with the link cost.
  - d. Run **commit**  
The configuration is committed.

----End

## (Optional) Configuring IPv4 IS-IS Attributes on Networks of Different Types

Different IS-IS attributes can be configured for different types of network interfaces.

### Context

The establishment mode of IS-IS neighbor relationships on a broadcast network is different from that on a P2P network. Different IS-IS attributes can be configured for interfaces on different types of networks.

IS-IS is required to select a DIS on a broadcast network. Configure the DIS priorities of IS-IS interfaces so that the interface with the highest priority is selected as the DIS.

The network types of the IS-IS interfaces on both ends of a link must be the same, otherwise, the IS-IS neighbor relationship cannot be established between the two interfaces. If the type of an interface on the neighbor is P2P, you can configure the interface type on the local device to P2P so that an IS-IS neighbor relationship can be established between the two devices.

IS-IS on a P2P network is not required to select a DIS. Therefore, you do not need to configure DIS priorities. To ensure the reliability of P2P links, configure IS-IS to use the three-way handshake mode for IS-IS neighbor relationship establishment so that faults on a unidirectional link can be detected.

### Procedure

- Configure the DIS priority of an IS-IS interface.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface interface-type interface-number**  
The interface view is displayed.

c. Run **isis dis-priority priority [ level-1 | level-2 ]**

The DIS priority is configured on the interface. The greater the value, the higher the priority.

d. (Optional) Run **isis dis-name symbolic-name**

A name is configured for the DIS to facilitate maintenance and management.

e. Run **commit**

The configuration is committed.

- Configure the network type of an IS-IS interface.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis circuit-type p2p**

The network type of the interface is set to P2P

When the network type of an IS-IS interface changes, interface configurations change accordingly.

- After a broadcast interface is configured as a P2P interface using the **isis circuit-type p2p** command, the default values of the interval at which Hello packets are sent, the number of Hello packets that IS-IS fails to receive from a neighbor before declaring the neighbor Down, the interval LSPs are retransmitted on a P2P link, and various IS-IS authentication modes take effect. Consequently, other configurations such as the DIS priority, DIS name, and interval at which CSNPs are sent on a broadcast network become invalid.

- After the **undo isis circuit-type** command is run to restore the network type, the default values of the interval at which Hello packets are sent, the number of Hello packets that IS-IS fails to receive from a neighbor before declaring the neighbor Down, the interval LSPs are retransmitted on a P2P link, the IS-IS authentication mode, the DIS priority, and the interval at which CSNPs are sent on a broadcast network take effect.

d. Run **commit**

The configuration is committed.

- Set the negotiation mode for the establishment of neighbor relationships over P2P links.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis ppp-negotiation { 2-way | 3-way [ only ] }**

A negotiation mode is specified for the interface.

This command applies only to neighbor relationship establishment over P2P links. In the case of a broadcast link, you can run the **isis circuit-type p2p** command to set the link type to P2P, and then set a negotiation mode for neighbor relationship establishment.

d. Run **commit**

The configuration is committed.

- Configure OSICP negotiation check on PPP interfaces.
  - a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis ppp-osicp-check**

The OSICP negotiation status is checked on a PPP interface.

This command is applicable only to PPP interfaces. For P2P interfaces, this command is invalid.

After this command is run, OSI network negotiation status of PPP affects IS-IS interface status. When PPP detects that the OSI network fails, the link status of the IS-IS interface goes down and the route to the network segment where the interface resides is not advertised through LSPs.

d. Run **commit**

The configuration is committed.

- Configure the scale of the Hello packets sent on the IS-IS interface.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

- c. Configure the size of the Hello packets to be sent by the IS-IS interface. Perform either of the following configurations as required:

- To configure the interface to send Hello packets without the padding field, run the **isis small-hello** command.
- To configure the interface to send standard Hello packets with the padding field, run the **isis padding-hello** command.

d. Run **commit**

The configuration is committed.

----End

## (Optional) Configuring IS-IS to Adjust the Flooding Rate (IPv4)

### Context

In a scenario where a large number of LSPs need to be flooded at a time, for example, in a large-scale network, because the maximum number of LSPs that can be sent by IS-IS per second is limited, the time needed to complete the LSP flooding at a time may exceed the expected time, which affects the convergence efficiency on the entire network. In this case, you can increase the maximum IS-IS LSP flooding rate by increasing the maximum number of LSPs that can be sent per second during IS-IS LSP flooding. This speeds up IS-IS LSP flooding and network convergence. When the flooding pressure on the network is heavy and flow control is required, you can reduce the maximum number of LSPs that can be sent by IS-IS per second to slow down the IS-IS LSP flooding rate and relieve the flooding pressure on nodes.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **isis lsp flood-control max-count *max-count-value***

The maximum number of LSPs that IS-IS sends every second is set.

#### Step 3 Run **commit**

The configuration is committed.

----End

## (Optional) Enabling LSP Fragment Extension on an IS-IS Device (IPv4)

If the LSP capacity is insufficient, newly imported routes and new TLVs fail to be added to LSP fragments. In this case, you can use LSP fragment extension to increase the LSP capacity, restoring the LSP space. When the LSP space is restored, the system automatically attempts to re-add these routes and TLVs to LSP fragments.

### Context

The **lsp-fragment-extend** command enables LSP fragment extension on an IS-IS device in a specified mode and at a specified level. An LSP fragment number occupies only one byte and therefore a maximum of 256 fragments are supported. If there is a large amount of LSP content and the number of fragments exceeds 256, some information is lost. LSP fragment extension can address such a problem. You can run the **virtual-system** command to configure one or more virtual systems to support more than 256 LSP fragments.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

An IS-IS process is created, and the IS-IS view is displayed.

**Step 3** Run **lsp-fragments-extend [ [ level-1 | level-2 | level-1-2 ] | [ mode-1 | mode-2 ] ] \***

LSP fragment extension is enabled on an IS-IS device, and a specified mode and level are specified.

----End

**(Optional) Enabling a Device to Encapsulate Only One Interface IP Address in IS-IS LSPs (IPv4)**

To implement interworking between Huawei and non-Huawei devices, you need to enable the type-132 TLV in LSPs to carry the IP address of only one IS-IS interface on the Huawei device.

**Context**

By default, on Huawei devices, the type-132 TLV in LSPs carries the IP addresses of all IS-IS interfaces. However, on some non-Huawei devices, the type-132 TLV in LSPs carries the IP address of only one IS-IS interface. To implement interworking between Huawei devices and these non-Huawei devices, enable the type-132 TLV in LSPs to carry the IP address of only one IS-IS interface on the Huawei devices.

**Procedure**

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

An IS-IS process is created, and its view is displayed.

**Step 3** Run **advertise one-interface-address**

The type-132 TLV in LSPs is enabled to carry the IP address of only one IS-IS interface.

**Step 4** Run **commit**

The configuration is committed.

----End

**Verifying the Configuration**

After configuring basic IPv4 IS-IS functions, check information about IS-IS neighbors, interfaces, and routes.

**Prerequisites**

The configurations of basic IPv4 IS-IS functions are complete.

## Procedure

- Step 1** Run the **display isis name-table** [ *process-id* | **vpn-instance** *vpn-instance-name* ] command to check the mapping between the hostname of the local device and the system ID.
- Step 2** Run the **display isis peer** [ **verbose** ] [ *process-id* | **vpn-instance** *vpn-instance-name* ] command to check information about IS-IS neighbors.
- Step 3** Run the **display isis interface** [ **verbose** ] [ *process-id* | **vpn-instance** *vpn-instance-name* ] command to check information about IS-IS interfaces.
- Step 4** Run the **display isis route** [ *process-id* | **vpn-instance** *vpn-instance-name* ] [ **ipv4** ] [ **verbose** | [ **level-1** | **level-2** ] | *ip-address* [ *mask* | *mask-length* ] ] \* command to check information about IS-IS routes.

----End

### 1.1.8.2.4 Configuring IPv4 IS-IS Route Selection

Configuring IS-IS route selection can achieve refined control over route selection.

## Usage Scenario

After basic IPv4 IS-IS functions are configured, IS-IS routes will be generated, enabling communication between different nodes on a network.

If multiple routes are available, the route discovered by IS-IS may not be the expected one, which does not meet network planning requirements nor facilitate traffic management. To address this issue, configure IPv4 IS-IS route selection to implement refined control over route selection.

To implement refined control over IPv4 IS-IS route selection, perform the following operations:

- **Configure the costs for IPv4 IS-IS interfaces.**

#### NOTE

Changing the IS-IS cost for an interface can control route selection, but routes on the interface need to be recalculated if a network topology changes, especially on a large-scale network. In addition, the configuration result may not meet your expectation.

Therefore, configure IS-IS costs before configuring basic IS-IS functions.

- Configure IPv4 IS-IS route leaking.
- Configure rules for selecting equal-cost IPv4 IS-IS routes.
- Filter IPv4 IS-IS routes.
- Configure an overload bit for an IPv4 IS-IS device.
- Configure an IPv4 IS-IS interface to automatically adjust the link cost.

## Pre-configuration Tasks

Before configuring IPv4 IS-IS route selection, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.

- [Configure basic IPv4 IS-IS functions.](#)

## Configuring IPv4 IS-IS Route Leaking

Configuring IS-IS route leaking enables you to optimize IS-IS route selection on a two-level-area network.

### Context

If multiple Level-1-2 devices in a Level-1 area are connected to devices in the Level-2 area, a Level-1 LSP sent by each Level-1-2 device carries an ATT flag bit of 1. This Level-1 area will have multiple routes to the Level-2 area and other Level-1 areas.

By default, routes in a Level-1 area can leak to the Level-2 area so that Level-1-2 and Level-2 devices can learn about the topology of the entire network. Devices in a Level-1 area are unaware of the entire network topology because they only maintain LSDBs in the local Level-1 area. Therefore, a device in a Level-1 area can forward traffic to a Level-2 device only through the nearest Level-1-2 device. However, the used route may not be optimal.

To enable a device in a Level-1 area to select the optimal route, configure IPv4 IS-IS route leaking so that specified routes in the Level-2 area can leak to the local Level-1 area.

If you want the Level-2 area to know only some of the routes in the local Level-1 area, configure a policy so that only desired routes can leak to the Level-2 area.

### Procedure

- Configure route leaking from the Level-2 area to the Level-1 area.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Configure the routes in the Level-2 area and other Level-1 areas that meet the specified conditions to leak to the local Level-1 area.  
Run any of the following commands as required:
    - Based on the basic ACL:
      - 1) Run **import-route isis level-2 into level-1 [ filter-policy { acl-number | acl-name acl-name } | tag tag | no-sid ]**  
The device is configured to leak routes into the local Level-1 area based on the specified basic ACL.
      - 2) Run **quit**  
Return to the system view.
      - 3) Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**  
The ACL view is displayed.

4) Run **rule [ rule-id ] [ name rule-name ] { deny | permit }**  
[ fragment-type { fragment | non-fragment | non-subseq |  
fragment-subseq | fragment-spe-first } | source { source-ip-  
address { source-wildcard | 0 | src-netmask } | any } | time-range  
time-name | vpn-instance vpn-instance-name ] \*

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- Based on an IP prefix list:

Run **import-route isis level-2 into level-1 [ filter-policy ip-prefix ip-prefix-name | tag tag | no-sid ] \***

The device is configured to leak routes into the local Level-1 area based on the specified IP prefix list.

- Based on a route-policy:

Run **import-route isis level-2 into level-1 [ filter-policy route-policy route-policy-name | tag tag | no-sid ] \***

The device is configured to leak routes into the local Level-1 area based on the specified route-policy.

 NOTE

The command is run on the Level-1-2 device that is connected to an external area.

d. Run **commit**

The configuration is committed.

- Configure route leaking from the Level-1 area to the Level-2 area.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

- Configure the routes that meet the specified conditions in the Level-1 area to leak to the Level-2 area.

Run any of the following commands as required:

■ Based on the basic ACL:

- Run **import-route isis level-1 into level-2 [ filter-policy { acl-number | acl-name acl-name } | tag tag | no-sid ]**\*

The device is configured to leak routes into the local Level-2 area based on the specified basic ACL.

- Run **quit**

Return to the system view.

- Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The ACL view is displayed.

- Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ]**\*

The rule for the ACL is configured.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.

- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- Based on an IP prefix list:

Run **import-route isis level-1 into level-2 [ filter-policy ip-prefix ip-prefix-name | tag tag | no-sid ] \***

The device is configured to leak routes into the local Level-2 area based on the specified IP prefix list.

- Based on a route-policy:

Run **import-route isis level-1 into level-2 [ filter-policy route-policy route-policy-name | tag tag | no-sid ] \***

The device is configured to leak routes into the local Level-2 area based on the specified route-policy.

#### NOTE

The command is run on the Level-1-2 device that is connected to an external area.

#### d. Run **commit**

The configuration is committed.

----End

## Configuring a Method for IS-IS to Process Equal-Cost Routes (IPv4)

If multiple equal-cost routes are available on an IS-IS network, you can configure load balancing to increase the bandwidth usage of each link, or configure weights for the equal-cost routes to facilitate service traffic management.

## Context

If there are multiple redundant links on an IS-IS network, multiple equal-cost routes may exist. In this case, you can use either of the following methods:

- Configure load balancing so that traffic is balanced among links.  
This method improves link utilization and reduces the possibility of congestion caused by link overload. However, load balancing randomly forwards traffic, which may affect service traffic management.
- Configure weights for equal-cost routes so that the route with the highest priority is preferentially selected, with others functioning as backups.  
In this mode, you can allow one or more routes to be preferentially selected without modifying the original configuration. This ensures network reliability and facilitates service traffic management.

## Procedure

- Configure load balancing for equal-cost IS-IS routes.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **maximum load-balancing number**  
The maximum number of equal-cost IS-IS routes for load balancing is set.

### NOTE

If the actual equal-cost routes outnumber the value specified in the **maximum load-balancing** command, routes are selected for load balancing based on the following rules:

1. Route weight: Routes with small weight values (high priority) are selected for load balancing. For details about route preference configuration, see [Configure priorities for equal-cost IS-IS routes](#).
2. Next-hop system ID: If routes have the same weight, those with small system IDs are selected for load balancing.
3. Outbound interface index: If routes have the same weight and system ID, those with small outbound interface indexes are selected for load balancing.
4. Next-hop IP address: If the weights, next-hop system IDs, and interface indexes of the routes are the same, their next-hop IP addresses are compared. The routes with high IP addresses are selected for load balancing.

- d. (Optional) Run **ecmp-prefer [ te-tunnel | intact ]**

The priority is set for the routes with a TE tunnel interface or an IPv4 interface as the outbound interface.

If both an IGP-Shortcut-enabled TE tunnel and IP link are available, you can configure a priority for the routes with a TE tunnel interface or an IPv4 interface as the outbound interface for route selection.

- e. Run **commit**  
The configuration is committed.
- Configure weights for equal-cost IS-IS routes.
  - a. Run **system-view**  
The system view is displayed.

- b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
- c. Run **nexthop ip-address weight value**  
A weight is configured for an equal-cost route.

 NOTE

A smaller *value* indicates a higher priority.

- d. Run **commit**  
The configuration is committed.

----End

## Filtering IPv4 IS-IS Routes

If some IS-IS routes are not preferred, configure conditions to filter IS-IS routes. Only IS-IS routes meeting the specified conditions can be added to an IP routing table.

### Context

Only routes in an IP routing table can be used to forward IP packets. An IS-IS route can take effect only after it has been added to an IP routing table.

You can configure a basic ACL, an IP prefix list, or a route-policy to filter routes so that only the matched IS-IS routes can be delivered to the IP routing table. IS-IS routes that do not meet the specified conditions cannot be added to the IP routing table nor selected to forward IP packets.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

#### Step 3 Configure IS-IS to deliver specified IS-IS routes to the IP routing table.

Run any of the following commands as required:

- Based on the basic ACL:
  - a. Run the **filter-policy { acl-number | acl-name acl-name } import** command.
  - b. Run **quit**  
Return to the system view.
  - c. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The ACL view is displayed.

- d. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.  
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.
- Based on an IP prefix list:  
Run **filter-policy ip-prefix ip-prefix-name import**
- Based on a route-policy:  
Run **filter-policy route-policy route-policy-name import**

#### Step 4 Run commit

The configuration is committed.

----End

## Configuring an Overload Bit for an IPv4 IS-IS Device

If an IS-IS device needs to be temporarily isolated, configure the overload state for it to prevent other devices from forwarding traffic to this IS-IS device and prevent routing black holes.

### Context

If an IS-IS device needs to be temporarily isolated, for upgrade or maintenance purposes for example, configure the overload state for it so that no other devices will forward traffic to this IS-IS device.

IS-IS routes converge more quickly than BGP routes do. To prevent routing black holes on a network where both IS-IS and BGP are configured, set an overload bit to instruct an IS-IS device to enter the overload state during its start or restart. After BGP convergence is complete, cancel the overload bit.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

#### Step 3 Run **set-overload { on-startup [ timeout1 | start-from-nbr system-id [ timeout1 [ timeout2 ] ] | wait-for-bgp [ timeout1 ] ] [ route-delay-distribute timeout4 ] [ send-sa-bit [ timeout3 ] ] [ route-max-metric ] } [ allow { interlevel | external } \* ]**

The overload bit is configured.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring IS-IS to Generate IPv4 Default Routes

This section describes how to configure IS-IS to generate default routes to control the advertising of IS-IS routing information.

### Context

The destination address and mask of a default route are all 0s. If the destination address of a packet does not match any entry in the routing table of a device, the device sends the packet along the default route. If neither the default route nor the destination address of the packet exists in the routing table, the device discards the packet and informs the source end that the destination address or network is unreachable.

IS-IS can generate default routes using either of the following mode:

- **Command-triggered default route generation mode**

You can run the **default-route-advertise** command on a device so that the device adds a default route to the LSP before sending the LSP to a neighbor. Therefore, the neighbor can learn this default route.

- **ATT bit 1-triggered default route generation mode**

According to IS-IS, a Level-1-2 router sets the ATT bit to 1 in the LSP to be advertised to a Level-1 area if the Level-1-2 router can reach more Level-1 areas through the Level-2 area than through the Level-1 area. After a Level-1 router in the Level-1 area receives the LSP, it generates a default route destined for the Level-1-2 router. Based on the network requirements, you can configure whether the Level-1-2 router sets the ATT bit carried in the LSP and whether a Level-1 router generates a default route after it receives the LSP carrying ATT bit 1.



This mode applies only to Level-1 routers.

## Procedure

- Configure command-triggered default route generation mode.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Run **default-route-advertise [ always | match default | route-policy route-policy-name | route-filter route-filter-name ] [ [ cost cost ] | [ tag tag ] | [ level-1 | level-2 | level-1-2 ] ] \* [ avoid-learning ]**

IS-IS is configured to generate default routes.

The IS-IS level of a router determines the IS-IS level of the generated default routes. The default routes generated using this command are advertised only to routers of the same level. You can configure a routing policy so that IS-IS generates default routes only when there are matched routes in the routing table.
  - d. Run **commit**

The configuration is committed.
- Configure ATT bit 1-triggered default route generation mode.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Run the following command as required:
    - To set the ATT bit in the LSPs sent by the Level-1-2 router, run the **attached-bit advertise { always | never }** command.
      - If the **always** parameter is specified, the ATT bit is set to 1. After receiving the LSPs carrying the ATT bit 1, the Level-1 router generates a default route.

- If the **never** parameter is specified, the ATT bit is set to 0. After receiving the LSPs carrying the ATT bit 0, the Level-1 router does not generate a default route, which reduces the size of a routing table.
  - To disable the Level-1 router from generating default routes even though it receives the LSPs carrying ATT bit 1, run the **attached-bit avoid-learning** command.
- d. Run **commit**

The configuration is committed.

----End

## Configuring an IPv4 IS-IS Interface to Automatically Adjust the Link Cost

Configuring an IS-IS interface to automatically adjust the link cost based on link quality facilitates route selection control and improves network reliability.

### Context

A bit error refers to the deviation between a bit that is sent and the bit that is received. The bit error rate (BER) refers to the number of bit errors divided by the total number of bits transferred during a studied time interval. During data transmission, a high BER may degrade or even interrupt services.

To prevent this problem, configure IS-IS interfaces to automatically adjust link costs based on link quality so that unreliable links are not used by the optimal routes.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 3 Run **isis enable [ process-id ]**

IS-IS is enabled on the interface.

#### Step 4 Run **isis link-quality low incr-cost { cost-value | max-reachable }**

The IS-IS interface is configured to automatically adjust the link cost based on link quality.

### NOTE

The *cost* parameter specifies the link cost adjustment value. After this parameter is specified:

- If the link quality changes to **low**, the link cost equals the sum of the original link cost and the incremental cost. If the sum exceeds the maximum link cost allowed, the maximum link cost allowed is used.
  - The maximum link cost is 63, if the cost style is narrow, narrow-compatible, or compatible.
  - The maximum link cost is 16777214, if the cost style is wide or wide-compatible.
- When the link quality recovers from **low**, the link cost of the interface is restored to the original value (the one before *cost* was added).

### Step 5 Run commit

The configuration is committed.

----End

## Configuring IPv4 IS-IS Route Recursion to IPv6 Next Hops

Configuring IPv4 IS-IS route recursion to IPv6 next hops allows IPv4 routes to be forwarded on IPv6 networks, improving network compatibility.

### Context

During the evolution from IPv4 to IPv6, some IPv4 services may fail to adapt to IPv6 in a short period of time. To ensure compatibility with these IPv4 services on the IPv6 network, you can configure IPv4 route recursion to IPv6 next hops.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run isis [ process-id ]

The IS-IS view is displayed.

#### Step 3 Run ipv4-prefix ipv6-nexthop enable

IPv4 route recursion to IPv6 next hops is configured.

#### Step 4 Run commit

The configuration is committed.

----End

## Verifying the IPv4 IS-IS Route Selection Configuration

After configuring IPv4 IS-IS route selection, check the configurations.

### Procedure

- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv4 ] [ verbose | [ level-1 | level-2 ] | ip-address [ mask | mask-length ] ]** \* command to check IS-IS routing information.

- Run the **display isis lsdb [ { level-1 | level-2 } | verbose | { local | lsp-id | is-name symbolic-name } ] \* [ process-id | vpn-instance vpn-instance-name ]** command to check information in the IS-IS LSDB.

----End

### 1.1.8.2.5 Configuring IPv4 IS-IS to Interact with Other Routing Protocols

If other routing protocols are configured on an IS-IS network, configure IS-IS to interact with these protocols for communication between them.

#### Usage Scenario

If other routing protocols are configured on an IS-IS network, note the following issues:

- Priorities of IS-IS routes

If multiple routes to the same destination are discovered by different routing protocols running on the same device, the route discovered by the protocol with the highest priority is selected. For example, if both OSPF and IS-IS are configured, the route discovered by OSPF is used because OSPF enjoys a higher priority than IS-IS by default.

If you want the route discovered by IS-IS to be preferentially selected, configure a higher priority for the route.

- Communication between areas

If other routing protocols are configured on an IS-IS network, configure IS-IS to interact with those routing protocols so that IS-IS areas can communicate with non-IS-IS areas.

#### NOTE

The LSDBs of different IS-IS processes on a device are independent of each other. Therefore, each IS-IS process on the device considers routes of the other IS-IS processes as external routes.

To ensure successful traffic forwarding, configure IS-IS to import external routes on a device (such as a Level-1-2 IS-IS router) where external routes are configured. Such configuration enables all devices in IS-IS areas to learn external routes, implementing refined control over traffic forwarding.

To ensure successful forwarding of traffic destined for IS-IS areas, enable the other routing protocols to interact with IS-IS.

#### Pre-configuration Tasks

Before configuring IPv4 IS-IS to interact with other routing protocols, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic IPv4 IS-IS functions.**
- Configure basic functions of other routing protocols.

## Configuring a Preference Value for IPv4 IS-IS

If multiple routes to the same destination are discovered by different routing protocols, configure the highest priority value for IS-IS so that a route discovered by IS-IS can be selected preferentially.

### Context

If multiple routes to the same destination are discovered by different routing protocols running on the same device, the route discovered by the protocol with the highest priority is selected.

For example, if both OSPF and IS-IS are configured on a network, the route discovered by OSPF is used because OSPF has a higher priority than IS-IS by default.

You can set a preference value for IS-IS to increase the priority of IS-IS routes so that they are preferentially selected. In addition, you can configure a route-policy to increase the priority of specified IS-IS routes, without affecting route selection.

### Procedure

- Configure a preference value for IS-IS.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **preference *preference***  
The IS-IS priority value is configured.
- Configure a preference value for specified IS-IS routes.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **preference { { route-policy *route-policy-name* | route-filter *route-filter-name* } | *preference* } \***  
A preference value is configured for the matched IS-IS routes.



*preference* takes effect only on the matched IS-IS routes.

d. Run **commit**

The configuration is committed.

----End

## Configuring IPv4 IS-IS to Import External Routes

Configuring IS-IS to import external routes on a boundary device enables devices in the IS-IS domain to learn external routes and guide traffic forwarding.

### Context

After IS-IS is configured on a boundary device to advertise a default route, the traffic destined for a destination outside the domain can be diverted to the boundary device for processing. However, because other devices in the IS-IS domain do not have external routes, a large amount of traffic is forwarded to the boundary device, which overloads the boundary device.

If multiple boundary devices are deployed, optimal routes to other routing domains need to be selected. In this case, all the other devices in the IS-IS routing domain must learn all or some external routes.

Routing policies can be configured to import only the external routes that match filtering conditions or advertise only the imported routes that match filtering conditions to other IS-IS devices.

#### NOTICE

IS-IS and other dynamic routing protocols such as OSPF and BGP often import routes from each other. If no routing policy is configured or a routing policy is incorrectly configured on a device where IS-IS, OSPF, and BGP import routes from each other, a Layer 3 routing loop may occur due to a route selection result change. As a result, services are compromised. For details about the cause of the routing loop, see [Routing Loop Detection for Routes Imported to IS-IS](#).

### Procedure

- Configure IS-IS to import external routes.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Configure IS-IS to import external routes.
    - To import external routes to IS-IS and set a cost for them, run the **import-route { direct | static | unr | { ospf | rip | isis } [ process-id ] | bgp [ permit-ibgp ] } [ cost-type { external | internal } | cost cost | tag tag | route-policy route-policy-name | [ level-1 | level-2 | level-1-2 ] ] \* or import-route { ospf | isis } [ process-id ] [ cost-type { external | internal } | cost cost | tag tag | route-policy route-policy-name | [ level-1 | level-2 | level-1-2 ] | no-sid ] \* command.**

- To import external routes to IS-IS and keep the original costs, run the **import-route { { ospf | rip | isis } [ process-id ] | bgp [ permit-ibgp ] | direct } inherit-cost [ tag tag | route-policy route-policy-name | [ level-1 | level-2 | level-1-2 ] ] \* or import-route { ospf | isis } [ process-id ] inherit-cost [ { level-1 | level-2 | level-1-2 } | tag tag | route-policy route-policy-name | no-sid ] \* command.** The protocol from which routes are imported cannot be **static**.

 NOTE

IS-IS advertises all imported external routes to an IS-IS routing domain by default.

If only some imported external routes need to be advertised to the IS-IS routing domain, run the **filter-policy export** command to set a filtering policy.

d. Run **commit**

The configuration is committed.

- Configure IS-IS to advertise some external routes to an IS-IS routing domain.
  - a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Configure the device to advertise some external routes to the IS-IS routing domain.

Run any of the following commands as required:

- Based on the basic ACL:

1) Run **filter-policy { acl-number| acl-name acl-name } export [ protocol [ process-id ] ]**

The filtering policy used by IS-IS to advertise routes is configured.

2) Run **quit**

Return to the system view.

3) Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The ACL view is displayed.

4) Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.  
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- Based on an IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name export** [ *protocol* [ *process-id* ] ]

- Based on a route-policy:

Run **filter-policy route-policy route-policy-name export** [ *protocol* [ *process-id* ] ]

#### NOTE

After this command is run, only external routes that meet the specified conditions can be advertised to the IS-IS routing domain.

#### d. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of Interaction Between IPv4 IS-IS and Other Routing Protocols

After configuring IS-IS to import routes from other protocols, check the configurations.

## Procedure

- Run the **display isis lsdb [ { level-1 | level-2 } | verbose | { local | lsp-id | is-name symbolic-name } ] \* [ process-id | vpn-instance vpn-instance-name ]** command to check IS-IS LSDB information.
- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv4 ] [ verbose | [ level-1 | level-2 ] | ip-address [ mask | mask-length ] ] \*** command to check IS-IS routing information.
- Run the **display ip routing-table ip-prefix ip-prefix-name [ verbose ]** command to check the IP routing table.

----End

### 1.1.8.2.6 Adjusting the IPv4 IS-IS Route Convergence Speed

Accelerating IS-IS route convergence can improve the fault location efficiency and network reliability.

## Usage Scenario

The procedure for implementing IS-IS is as follows:

- Establishment of neighbor relationships: establishes neighbor relationships by exchanging Hello packets between two devices.
- LSP flooding: implements LSDB synchronization between devices in the same area.
- SPF calculation: uses the SPF algorithm to calculate IS-IS routes, and delivers IS-IS routes to the routing table.

To accelerate the IS-IS route convergence, configure the following parameters:

- Interval for detecting failures on IS-IS neighboring devices
- Flooding parameters of CSNPs and LSPs
- Interval for SPF calculation

You can also configure convergence priorities for IPv4 IS-IS routes so that key routes can converge first if the network topology changes, which minimizes adverse impacts on key services.

## Pre-configuration Tasks

Before adjusting the IPv4 IS-IS route convergence speed, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic IPv4 IS-IS functions.**

## Configuring the Interval for Detecting IS-IS Neighboring Device Failures

To minimize the impacts of neighboring device failures on IS-IS networks, accelerate the detection IS-IS neighboring device failures.

## Context

Connection status between an IS-IS device and its neighboring devices can be monitored by exchanging Hello packets at intervals. An IS-IS neighboring device is considered. Down if the IS-IS device does not receive any Hello packets from the neighboring device within a specified period (holding time). A failure in an IS-IS neighboring device will trigger LSP flooding and SPF calculation, after which IS-IS routes re-converge.

To adjust the fault detection speed, use the following methods to accelerate the speed of detecting neighboring device failures:

- Configure the interval at which Hello packets are sent.
- Configure the number of Hello packets that are sent before the local device considers the neighbor Down.

### NOTE

Holding time of neighboring devices = Interval at which Hello packets are sent x Number of Hello packets that are sent before the local device considers the neighbor Down. The maximum value of the holding time is 65535s.

- **Configure Dynamic BFD for IS-IS.**

### NOTE

Configuring IPv4 BFD for IS-IS is recommended because this method ensures faster fault detection than the other two methods.

## Procedure

- Set an interval at which Hello packets are sent.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface interface-type interface-number**  
The interface view is displayed.
  - c. Run **isis timer hello hello-interval [ level-1 | level-2 ] [ conservative ]**  
The interval at which Hello packets are sent is set.  
If the **conservative** parameter is specified in the command, the conservative mode is enabled for the holdtime of the IS-IS neighbor relationship.
    - If the parameter is specified and the holdtime of the IS-IS neighbor relationship is less than 20s, the IS-IS neighbor relationship is disconnected when the hold time elapses.
    - If the parameter is not specified and the holdtime of the IS-IS neighbor relationship is less than 20s, the IS-IS neighbor relationship is disconnected when the period of the hold time and a delay elapses.

 NOTE

A broadcast link can transmit both Level-1 and Level-2 Hello packets. You can set different intervals for these two types of Hello packets. By default, both Level-1 and Level-2 Hello packets are sent.

A P2P link can transmit only one type of Hello packets. Therefore, neither **level-1** or **level-2** needs to be specified if a P2P link is used.

The timer must be longer than the time a device takes to perform a master/slave main control board switchover. If the timer is set to less than the switchover time, a protocol intermittent interruption occurs during a switchover. The default timer value is recommended.

d. Run **commit**

The configuration is committed.

- Set the number of Hello packets that are sent before the local device considers the neighbor Down.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis timer holding-multiplier number [ level-1 | level-2 ]**

The number of Hello packets that are sent before the local device considers the neighbor Down is configured.

 NOTE

A broadcast link can transmit both Level-1 and Level-2 Hello packets. You can set different intervals for these two types of Hello packets. If no level is specified, the configured interval takes effect on both Level-1 and Level-2 Hello packets.

A P2P link can transmit only one type of Hello packets. Therefore, neither **level-1** or **level-2** needs to be specified if a P2P link is used.

d. Run **commit**

The configuration is committed.

----End

## Setting Flooding Parameters of SNPs and LSPs

To speed up LSDB synchronization between devices, set proper values for flooding parameters of SNPs and LSPs.

## Context

SNPs consist of CSNPs and PSNPs. CSNPs carry summaries of all LSPs in LSDBs, ensuring LSDB synchronization between neighboring routers. SNPs are processed differently on broadcast links and P2P links.

- On a broadcast link, CSNPs are periodically sent by a DIS device. If a router detects that its LSDB is not synchronized with that on its neighboring router, the router will send PSNPs to apply for missing LSPs.
- On a P2P link, CSNPs are sent only during initial establishment of neighbor relationships. If a request is acknowledged, a neighboring router will send a

PSNP in response to a CSNP. If a router detects that its LSDB is not synchronized with that on its neighboring router, the router will send PSNPs to apply for missing LSPs.

You can modify the following parameters of SNPs and LSPs on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X to speed up LSDB synchronization:

- **Set an interval at which CSNPs are sent.**
- **Configure the intelligent timer to control LSP generation.**
- **Set the size of an LSP.**
- **Set the LSP update interval.**
- **Set the maximum lifetime for LSPs.**
- **Set the maximum holdtime for the largest IS-IS route cost in local LSPs.**
- **Enable LSP fast flooding.**
- **Set an interval at which LSPs are retransmitted over a P2P link.**
- **Configure automatic IS-IS LSP Remaining Lifetime adjustment.**

## Procedure

- Set an interval at which CSNPs are sent.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface interface-type interface-number**  
The interface view is displayed.
  - c. Run **isis timer csnp csnp-interval [ level-1 | level-2 ]**  
The interval at which CSNPs are sent is set on the specified interface.
- Configure the intelligent timer to control LSP generation.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **timer lsp-generation max-interval [ init-interval [ incr-interval ] ] [ level-1 | level-2 ]**  
The intelligent timer is configured to control LSP generation.  
The intelligent timer involves three parameters, and the parameters are described as follows:
    - When only *max-interval* is used, the intelligent timer becomes a one-shot timer.

- If both *init-interval* and *incr-interval* are configured, the initial interval for generating an LSP is *init-interval*, and the interval for generating an LSP with the same LSP ID for the second time is *incr-interval*. From the third time on, each time the route changes, the interval for generating an LSP doubles until the interval reaches *max-interval*. If the local routing information still changes frequently within *max-interval*, the delay remains *max-interval*. If the local routing information does not change after the interval specified by *max-interval* expires or the IS-IS process is restarted, the interval decreases to *init-interval*.
  - If *init-interval* is specified but *incr-interval* is not specified, *init-interval* is used as the interval for generating an LSP for the first time, and *max-interval* is used as the interval for generating subsequent LSPs. If the local routing information changes frequently within the interval specified by *max-interval*, the delay remains *max-interval*. If the local routing information does not change after the interval specified by *max-interval* expires or the IS-IS process is restarted, the interval decreases to *init-interval*.
- d. Run **suppress-flapping lsp-generation timer** *delay-interval* [ **threshold** *threshold-value* ]
- A period is specified for the system to delay generating the same LSP during route flapping.
- e. Run **commit**
- The configuration is committed.
- Set the size of an LSP.
    - a. Run **system-view**
    - The system view is displayed.
    - b. Run **isis** [ *process-id* ]
    - The IS-IS view is displayed.
    - c. Run **lsp-length originate** *max-size*
    - The size of an LSP to be generated is set.
    - d. Run **lsp-length receive** *max-size*
    - The size of an LSP to be received is set.

 **NOTE**

*max-size* of an LSP to be generated must be less than or equal to *max-size* of an LSP to be received.

The value of *max-size* set using the **lsp-length** command must meet the following requirements; otherwise, the MTU status on the interface is considered down.

- The MTU of an Ethernet interface must be greater than or equal to the sum of the value of *max-size* plus 3.
- The MTU of a P2P interface must be greater than or equal to the value of *max-size*.

e. Run **commit**

The configuration is committed.

- Set the LSP update interval.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run **timer lsp-refresh refresh-value**

An LSP update interval is set.

To ensure the synchronization of LSPs in the entire area, IS-IS periodically sends all the current LSPs.

 NOTE

Ensure that the LSP update interval is at least 300s shorter than the maximum LSP lifetime so that new LSPs can reach all routers in the area before the original LSPs expire.

The larger a network, the greater the deviation between the LSP update interval and the maximum LSP lifetime.

d. Run **commit**

The configuration is committed.

- Set the maximum lifetime for LSPs.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run **timer lsp-max-age max-age-value**

The maximum lifetime is set for LSPs.

When a device generates the system LSP, it fills in the maximum lifetime for this LSP. The lifetime of the LSP decreases with time. If the device does not receive any updated LSPs and the lifetime of the LSP is reduced to 0, the device keeps the LSP for another 60s. If the device fails to receive any updated LSPs within the 60s, it deletes the LSP from the LSDB.

d. Run **commit**

The configuration is committed.

- Set the maximum holdtime for the largest IS-IS route cost in local LSPs.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis [ process-id process-id ] peer hold-max-cost timer timer**

The holdtime of the maximum cost in local IS-IS LSPs is set.

When an IS-IS interface changes from down to up, the IS-IS neighbor relationship is re-established. After IGP route convergence is completed, traffic is switched back. In most cases, IGP routes converge quickly.

However, many services that depend on IGP routes may require a delayed switchback. To achieve this, run the **isis peer hold-max-cost** command to set the holdtime for the maximum route cost (16777214 in wide cost style and 63 in narrow cost style) in local IS-IS LSPs, so that traffic is forwarded over the original path before the hold-max-cost timer expires. After the timer expires, the normal cost value is restored, and then traffic is normally switched back.

d. (Optional) Run **isis peer hold-cost cost-val timer timer-val**

The period during which IS-IS keeps the specified cost in local LSPs is set.

e. Run **commit**

The configuration is committed.

● Enable LSP fast flooding.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis timer lsp-throttle throttle-interval [ count count ]**

The minimum interval at which LSPs are sent by the interface and the maximum number of LSPs that can be sent within the interval are set.

The *count* parameter specifies the maximum number of LSPs that can be sent within the interval specified by *throttle-interval*. The value is an integer ranging from 1 to 1000.

d. Run **isis [ process-id ]**

The IS-IS view is displayed.

e. Run **suppress-flapping lsp-flood timer delay-interval [ threshold threshold-value ]**

A period is specified for the system to delay LSP flooding during route flapping.

f. Run **commit**

The configuration is committed.

● Set an interval at which LSPs are retransmitted over a P2P link.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

- c. (Optional) Run **isis circuit-type p2p [ strict-snpa-check ]**

The broadcast interface is simulated as a P2P interface.

An interval at which LSPs are retransmitted takes effect only on P2P interfaces. Therefore, to configure the interval on a broadcast interface, change the broadcast interface to a P2P interface first.

- d. Run **isis timer lsp-retransmit *retransmit-interval***

The interval at which LSPs are retransmitted over a P2P link is set.

- e. Run **commit**

The configuration is committed.

- Configure automatic IS-IS LSP Remaining Lifetime adjustment.

- a. Run **system-view**

The system view is displayed.

- b. Run **isis [ *process-id* ]**

The IS-IS view is displayed.

- c. (Optional) Run **undo lsp-remaining-lifetime refresh disable**

Automatic IS-IS LSP Remaining Lifetime adjustment is enabled.

- d. Run **lsp-remaining-lifetime refresh timer { *refreshvalue* | lsp-max-age }**

An IS-IS LSP Remaining Lifetime value is set.

- e. Run **commit**

The configuration is committed.

----End

## Adjusting the SPF Calculation Interval

By adjusting the SPF calculation interval, you can ensure that IS-IS responds to network changes in time and reduce the system resources consumed by SPF calculation.

### Context

When a network changes frequently, IS-IS performs SPF calculation frequently. Frequent SPF calculations consume a large number of CPU resources, affecting services.

The advantage of configuring an intelligent timer is that the interval for SPF calculation is short, which speeds up IS-IS route convergence. When the topology of the entire IS-IS network becomes stable, the interval between two SPF calculations is prolonged to reduce unnecessary calculations.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

**Step 2 Run `isis [ process-id ]`**

The IS-IS view is displayed.

**Step 3 Run `timer spf max-interval [ init-interval [ incr-interval ] ]`**

The SPF intelligent timer is configured.

The interval for SPF calculation is described as follows:

- The delay for the first SPF calculation is *init-interval*; the delay for the second SPF calculation is *incr-interval*. From the third time on, the SPF calculation delay doubles each time the route changes until the delay reaches *max-interval*. If network flapping persists within the interval specified by *max-interval*, the delay remains *max-interval*. If the network does not flap within *max-interval* or the IS-IS process is restarted, the delay decreases to *init-interval*.
- If *incr-interval* is not used, the delay for the first SPF calculation is *init-interval*. From the second time on, the delay remains *max-interval*. If the local routing information changes frequently within *max-interval*, the delay for SPF calculation remains *max-interval*. If the local routing information does not change within the interval specified by *max-interval* or the IS-IS process is restarted, the delay decreases to *init-interval*.
- When only *max-interval* is used, the intelligent timer becomes a one-shot timer.

**Step 4 Run `suppress-flapping route-calculate timer delay-interval [ threshold threshold-value ]`**

A period is specified for the device to delay route calculation when the device receives a purge LSP.

**Step 5 Run `timer purge-zero-lsp route-calculate-delay delay-interval`**

A period is specified for the device to delay route calculation when the device receives a purge LSP.

**Step 6 Run `commit`**

The configuration is committed.

----End

## Configuring Convergence Priorities for IPv4 IS-IS Routes

You can set a high convergence priority for key routes on an IS-IS network to ensure that these routes converge first if the network topology changes. This minimizes the impact on important services.

## Context

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X allows you to configure the highest convergence priority for specific IS-IS routes so that those IS-IS routes converge first when a network topology changes.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run isis [*process-id*]

The IS-IS view is displayed.

### Step 3 Run prefix-priority [ level-1 | level-2 ] { critical | high | medium | very-low } { ip-prefix *prefix-name* | tag *tag-value* }

A convergence priority is set for IS-IS routes.

The rules for applying the convergence priorities of IS-IS routes are as follows:

- Existing IS-IS routes converge based on the priorities configured in the **prefix-priority** command.
- New IS-IS routes that match the filtering rules converge based on the priorities configured in the **prefix-priority** command.
- If an IS-IS route meets the matching rules of multiple convergence priorities, the highest convergence priority is used.
- The convergence priority of Level-1 IS-IS routes is higher than that of Level-2 IS-IS routes.
- If no level is specified, the configuration takes effect on both Level-1 and Level-2 IS-IS routes.

#### NOTE

The **prefix-priority** command is only applicable to the public network.

After the **prefix-priority** command is run, the convergence priority of 32-bit host routes is **low**, and the convergence priorities of the other routes are specified in the **prefix-priority** command.

### Step 4 Run commit

The configuration is committed.

----End

## Enabling IS-IS Intelligent Convergence

Enabling IS-IS intelligent convergence can speed up IS-IS route convergence, thereby improving convergence performance.

## Context

In a fault-triggered switching scenario where the local device receives a route from a single node, IS-IS intelligent convergence can be enabled to allow IS-IS to perform fast route convergence by using the fast convergence algorithm. This improves convergence performance.

## Procedure

### Step 1 Run system-view

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

The IS-IS view is displayed.

**Step 3** Run **intelligent-convergence enable**

IS-IS intelligent convergence is enabled.

 **NOTE**

In the same networking scenario, the convergence speeds of devices with the **intelligent-convergence enable** command configuration increase significantly, far higher than those of the devices without this command configuration. As a result, routing loops may occur. Therefore, exercise caution when you run this command.

**Step 4** Run **commit**

The configuration is committed.

----End

## Verifying the IPv4 IS-IS Route Convergence Speed Configuration

After configuring parameters to adjust the IPv4 IS-IS route convergence speed, check the configurations.

## Procedure

- Run the **display isis interface [ [ verbose | traffic-eng ] \* | tunnel ] [ process-id | vpn-instance vpn-instance-name ]** command to check IS-IS packet information.
- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv4 ] [ topology topology-name ] [ verbose | [ level-1 | level-2 ] | ip-address [ mask | mask-length ] ] \***  command to check the priority of IS-IS routes.

----End

### 1.1.8.2.7 Configuring an IS-IS Multi-Instance Process

To reduce the interface count and configuration workload required by multiple conventional IS-IS processes, you can configure IS-IS multi-instance processes. A conventional IS-IS process and multiple IS-IS multi-instance processes can be configured on the same interface.

## Usage Scenario

On IS-IS networks, multiple IS-IS processes need to be used to isolate different access rings. To close the access rings, the IS-IS processes on all access rings need to be enabled. In this case, you need to enable different IGP processes on one interface. This reduces the interface count and configuration workload of the access rings.

## Pre-configuration Tasks

Before configuring an IS-IS multi-instance process, complete the following tasks:

- Configure a link layer protocol.
- Configure IP addresses for interfaces and ensure that neighboring devices are reachable at the network layer.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis [ process-id ]**

An IS-IS process is created, and its view is displayed.

*process-id* specifies the ID of an IS-IS process. If *process-id* is not specified, 1 is used as the IS-IS process ID. To associate an IS-IS process with a VPN instance, run the **isis process-id vpn-instance vpn-instance-name** command.

### Step 3 Run **multi-instance enable iid iid-value**

The IS-IS process is configured as an IS-IS multi-instance process.

### Step 4 Run **quit**

Return to the system view.

### Step 5 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 6 Run the **isis enable [ process-id ]** command to enable IS-IS on the interface and associate the IS-IS process with the interface or run the **isis ipv6 enable process-id** command to enable IS-IS IPv6 on the interface and associate the IS-IS process with the interface.

### Step 7 Configure parameters for the IS-IS multi-instance process as required.

- Run the **isis process-id process-id authentication-mode** command to configure the IS-IS interface to authenticate Hello packets using the specified authentication mode and password.
- Run the **isis process-id process-id cost** command to set a link cost for the IS-IS interface.
- Run the **isis process-id process-id ipv6 cost** command to set a link cost in the IPv6 topology.
- Run the **isis process-id process-id circuit-type** command to simulate an IS-IS broadcast interface as a P2P interface.
- Run the **isis process-id process-id prefix-sid** command to configure the IP address of a loopback interface as a Segment Routing prefix segment ID (SID).

### Step 8 Run **commit**

The configuration is committed.

For details on how to configure other functions for an IS-IS process, see [Creating an IPv4 IS-IS process](#) or [Creating an IPv6 IS-IS process](#).

----End

## Checking the Configurations

After configuring an IS-IS multi-instance process, check the configurations.

- Run the **display isis interface *interface-type* *interface-number* [ verbose ]** command to check information about the IS-IS interface. The command output shows more than one IS-IS process on the interface.

### 1.1.8.2.8 Enabling the Advertisement of IPv4 Delay Information

To allow IS-IS to collect and flood information about the intra-area IPv4 link delay, you can enable the advertisement of IPv4 delay information for an IS-IS process.

## Usage Scenario

Traditional path computation algorithms compute the optimal path to a destination address based on costs. However, such a computed path may not have the minimum delay. For services that have stringent requirements on delay, path computation can be performed based on the delay rather than on the cost to ensure that service traffic is transmitted along the path with the minimum delay. To meet these requirements, enable the advertisement of IPv4 delay information for an IS-IS process. After this function is enabled, IS-IS collects and floods information about the intra-area IPv4 link delay, and BGP-LS reports the information to the controller. This allows the controller to use the delay information to compute the optimal path on the P2P network.

## Pre-configuration Tasks

Before enabling the advertisement of delay information, complete the following task:

- Configure the TWAMP Light controller to detect delay information.
- Set the network type of the IS-IS interface to P2P.**

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run isis [ *process-id* ]

An IS-IS process is created, and the view is displayed.

*process-id* specifies an IS-IS process ID. If the *process-id* parameter is not specified, the system creates process 1 by default. To bind an IS-IS process to a VPN instance, run the **isis *process-id* vpn-instance *vpn-instance-name*** command.

### Step 3 Run cost-style { wide | wide-compatible | compatible }

A cost style is set for IS-IS.

### Step 4 (Optional) Run metric-delay normalize interval *interval-value* [ offset *offset-value* ]

The link delay normalization function is configured for the IS-IS process.

For the delay-based path computation algorithm, the delay differences of links are different, and the differences may be small. However, even if the delay differences are small, only one optimal path can be generated according to the existing SPF algorithm, and load balancing cannot be implemented within a certain delay tolerance range. As a result, link resources on the network cannot be fully utilized. To resolve this problem to the greatest extent, normalization processing may be performed on the link delays with a small difference or a difference within an acceptable range so that load balancing can be implemented and link resources on the network can be fully utilized.

You can also run the **isis [ process-id process-id-value ] metric-delay normalize interval interval-value [ offset offset-value ]** command in the IS-IS interface view to configure the link delay normalization function on an interface. If the function is configured both for an IS-IS process and an IS-IS interface, the interface's configuration takes precedence over the process's configuration. Delay variation cannot be normalized.

**Step 5** Run **traffic-eng [ level-1 | level-2 | level-1-2 ]**

TE is enabled on a specified level for the IS-IS process.

**Step 6** Run **metric-delay advertisement enable [ level-1 | level-2 | level-1-2 ]**

The advertisement of the maximum and minimum delay information is enabled.

**Step 7** Run **metric-delay average advertisement enable [ level-1 | level-2 | level-1-2 ]**

The advertisement of the average delay is enabled.

**Step 8** Run **metric-delay variation advertisement enable [ level-1 | level-2 | level-1-2 ]**

The advertisement of delay variation is enabled.

**Step 9** (Optional) Run **metric-delay suppress timer time-value percent-threshold percent-value absolute-threshold absolute-value**

Parameters used to suppress the advertisement of delay information are configured.

**Step 10** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display isis interface verbose traffic-eng** command to check information about IS-IS-enabled interfaces.
- Run the **display isis traffic-eng advertisements** command to check the TE information advertised by IS-IS.

### 1.1.8.2.9 Configuring Interface MSD Advertisement (IPv4)

Interface maximum SID depth (MSD) advertisement allows IS-IS to advertise the MSD of Segment Routing (SR) interfaces.

## Usage Scenario

When calculating paths for SR-TE Policies in an SR scenario, the controller needs to determine paths and calculate binding SIDs based on the MSD advertised by an IGP. When the link MSDs of multiple interfaces on a device are different, the node MSD advertised by IS-IS is the smallest value among all link MSDs. Comparing with advertising node MSDs only, advertising link MSDs provides more accurate MSD information, allowing the controller to calculate paths more efficiently.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **segment-routing**

The SR view is displayed.

### Step 3 Run **quit**

Return to the system view.

### Step 4 Run **isis [ process-id ]**

An IS-IS process is created, and the IS-IS view is displayed.

### Step 5 Run **cost-style { wide | wide-compatible | compatible }**

An IS-IS cost style is set.

### Step 6 Run **segment-routing mpls**

SR-MPLS is enabled for IS-IS.

### Step 7 Run **link-msd advertisement enable [ level-1 | level-2 | level-1-2 ]**

Interface MSD advertisement is enabled.

### Step 8 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display isis lsdb verbose** command to check IS-IS LSDB information.
- Run the **display isis traffic-eng advertisements** command to check the TE information advertised by IS-IS.

### 1.1.8.2.10 Configuring an IS-IS Process to Advertise IPv4 Packet Loss Rates

Configuring an IS-IS process to advertise IPv4 packet loss rates helps ensure that service traffic is transmitted along the path with the lowest packet loss rate.

## Prerequisites

Before configuring an IS-IS process to advertise packet loss rates, complete the following tasks:

- Run the **cost-style** command to set the IS-IS cost style to wide, wide-compatible, or compatible.
- Run the **traffic-eng** command to enable IS-IS TE.

## Context

Traditional path computation algorithms compute the optimal path to a destination address based on costs. However, this path may not be the one with the lowest packet loss rate. For services that have stringent requirements on the packet loss rate, path computation can be performed based on packet loss rates instead of costs to ensure that service traffic is transmitted along the path with the lowest packet loss rate. Specifically, configure an IS-IS process to advertise IPv4 packet loss rates. After this function is configured, IS-IS collects and floods information about intra-area IPv4 packet loss rates, and BGP-LS reports the information to the controller. The controller then computes the optimal path on the P2P network based on the packet loss rates.

## Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** Run the **isis [ process-id ]** command to create an IS-IS process and enter its view.
- Step 3** Run the **cost-style { wide | wide-compatible | compatible }** command to set an IS-IS cost style.
- Step 4** Run the **traffic-eng [ level-1 | level-2 | level-1-2 ]** command to enable IPv4 TE for a specified level of the IS-IS process.
- Step 5** Run the **metric-link-loss advertisement enable [ level-1 | level-2 | level-1-2 ]** command to configure the function of advertising IPv4 packet loss rates.
- Step 6** (Optional) Run the **metric-link-loss suppress timer *timer-value* percent-threshold *percent-value* absolute-threshold *absolute-value*** command to configure suppression parameters for packet loss rate advertisement.

### NOTE

The advertised packet loss rates may be dynamic ones measured using TWAMP Light or static ones manually configured.

The valid dynamic packet loss rates measured using TWAMP Light are advertised only to the IS-IS interface bound to the TWAMP Light test session.

After receiving packet loss rate information, IS-IS advertises and floods the information.

- Step 7** Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

After configuring the function, verify the configuration.

- Run the **display isis interface verbose traffic-eng** command to check information about IS-IS-enabled interfaces.
- Run the **display isis traffic-eng advertisements** command to check the TE information advertised by IS-IS.

### 1.1.8.2.11 Configuring LFA Link Attribute Advertisement

Configuring LFA link attribute advertisement in an IS-IS process helps collect and flood the SRLG information applied by LFA in a domain.

#### Prerequisites

Before configuring LFA link attribute advertisement, complete the following tasks:

- Run the **cost-style** command to set the IS-IS cost style to wide, wide-compatible, or compatible.

#### Context

Without the SRLG link attribute used by LFA on the remote device, the local device can only preferentially select a local link that is not in the same SRLG as the primary path when using IS-IS TI-LFA to calculate a backup path. If a remote link on the backup path is in the same SRLG as the primary path and the primary path fails, the backup path may also fail. After traffic is switched to the backup path, network traffic may be interrupted. To solve the preceding problem, you can configure the advertisement of SRLG link attribute information applied by LFA through IS-IS LSPs. When the local device uses TI-LFA to calculate a backup path, it can preferentially select a remote link that is not in the same SRLG as the local link as the backup path based on the LFA link attribute information advertised by the remote device, reducing the possibility of network traffic interruption.

#### Procedure

- Step 1 Run the **system-view** command to enter the system view.
- Step 2 Run the **isis [ process-id ]** command to create an IS-IS process and enter its view.
- Step 3 Run the **cost-style { wide | wide-compatible | compatible }** command to set an IS-IS cost style.
- Step 4 Run the **advertise link-attributes application lfa** command to configure LFA link attribute advertisement.
- Step 5 Run the **commit** command to commit the configuration.

----End

#### Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display isis lsdb verbose** command to check LFA link attributes.

### 1.1.8.2.12 Configuring IS-IS Extended Prefix Attribute Advertisement (IPv4)

After IS-IS extended prefix advertisement is configured, route origin information can be advertised to the LSDB so that the device can determine the origin of received routes.

#### Context

IS-IS defines a type of extended prefix attribute (IPv4/IPv6 Extended Reachability Attribute) sub-TLVs. These sub-TLVs are used to describe the origin of advertised routes and can be advertised in IPv4, IPv6, and locator TLVs. Based on the advertised sub-TLVs, the device can determine whether the received routes are inter-domain routes or host routes.

The extended prefix attribute flag (IPv4/IPv6 Extended Reachability Attribute Flags) is a part of the extended prefix attribute sub-TLV. You can run a command to determine whether this flag is advertised.

#### Pre-configuration Tasks

Before configuring IS-IS extended prefix attribute advertisement, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure addresses for interfaces to ensure that neighboring devices are reachable at the network layer.
- [Configure basic IPv4 IS-IS functions](#).

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **isis [ process-id ]**

An IS-IS process is created, and its view is displayed.

##### Step 3 Run **advertise prefix-attributes flags**

The IS-IS process is enabled to advertise the extended prefix attribute flag.

##### Step 4 Run **quit**

Return to the system view.

##### Step 5 (Optional) Set the N flag of the extended prefix attribute on a loopback interface in the IS-IS process to 0.

By default, if a loopback interface's route with a 32-bit subnet mask is a host route, the extended prefix N flag is set to 1. To set the extended prefix N flag to 0, perform this step.

1. Run the **interface Loopback interface-number** command to enter the loopback interface view.
2. Run the **isis enable [ process-id ]** command to enable IS-IS on the interface.

3. Run the **isis [ process-id process-id-value ] prefix-attributes node-disable** command to set the N flag of the extended prefix attribute on the loopback interface in the IS-IS process to 0.
4. Run the **quit** command to return to the system view.

#### Step 6 Run commit

The configuration is committed.

----End

### Verifying the Configuration

After the configuration is complete, run the **display isis lsdb verbose** command to view information about the extended prefix attribute flag.

#### 1.1.8.2.13 Configuring Static BFD for IS-IS

BFD can provide link fault detection featuring light load and high speed (within milliseconds). Static BFD needs to be configured.

### Usage Scenario

On IS-IS networks, IS-IS neighbors periodically send Hello packets to detect neighbor status changes. For networks that require fast convergence and zero packet loss, IS-IS is unreliable to detect link faults. To address this issue, configure BFD for IS-IS.

BFD includes static BFD and dynamic BFD. Static BFD is easy to control and flexible to deploy. To save memory and ensure the reliability of key links, implement BFD on these links. Static BFD helps detect link faults rapidly and implement fast route convergence.

To configure a static BFD session, you need to manually configure BFD session parameters, including the local and remote discriminators.

#### NOTE

BFD detects only the one-hop link between IS-IS neighbors because IS-IS establishes only one-hop neighbors.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bfd**

Global BFD is enabled on the local end.

#### Step 3 Run **quit**

Return to the system view.

#### Step 4 (Optional) To configure BFD session check for an IS-IS process, perform the following steps:

1. Run **isis [ process-id ]**  
The IS-IS view is displayed.
2. Run **bfd session-up check**

BFD session check is configured for the IS-IS process.

 NOTE

When the Layer 2 network is running normally, IS-IS neighbor relationships can be established and routes can be delivered. However, if the Layer 3 network is unreachable, upper-layer traffic loss occurs. To resolve this problem, configure BFD session check for an IS-IS process by running the **bfd session-up check** command. This ensures that an IS-IS neighbor relationship is established only when the BFD session is up on corresponding interfaces. After this function is configured, it applies only to the neighbor relationships to be established (it does not apply to existing neighbor relationships).

**Step 5** Run **interface interface-type interface-number**

The interface view is displayed.

BFD can be enabled on physical interfaces only.

**Step 6** Run **isis bfd static**

Static BFD is enabled on the interface.

**Step 7** Run **quit**

Return to the system view.

**Step 8** Run **bfd session-name bind peer-ip ip-address [ interface { interface-name | interface-type interface-number } ]**

A BFD session is bound to an interface.

If both **peer-ip** (IP address of the peer end) and **interface** (local interface) are specified, BFD only monitors a single-hop link with **interface** as the outbound interface and **peer-ip** as the next hop address.

**Step 9** Run the following commands as required:

- To set the local discriminator, run the **discriminator local descr-value** command.
- To set the remote discriminator, run the **discriminator remote descr-value** command.

 NOTE

The local discriminator of the local device must be the same as the remote discriminator of the peer device, and the remote discriminator of the local device must be the same as the local discriminator of the peer device.

**Step 10** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display isis bfd [ process-id | vpn-instance vpn-instance-name ] session { peer ip-address | all }** command to view the information about the BFD session.
- Run the **display isis interface verbose** command to view the configurations of BFD for IS-IS on an interface.

### 1.1.8.2.14 Configuring Dynamic BFD for IS-IS

BFD can provide link failure detection featuring light load and high speed (within milliseconds). With dynamic BFD, routing protocols can dynamically trigger the establishment of BFD sessions.

#### Usage Scenario

On IS-IS networks, IS-IS neighbors periodically send Hello packets to detect neighbor status changes. For networks that require fast convergence and zero packet loss, IS-IS is unreliable to detect link faults. To address this issue, configure BFD for IS-IS.

BFD includes static BFD and dynamic BFD. In dynamic BFD, BFD session establishment is triggered by routing protocols. Dynamic BFD minimizes configuration errors caused by manual operations in static BFD and is easy to configure. Therefore, dynamic BFD is applicable to networks on which all devices require BFD. Dynamic BFD helps detect link faults rapidly and implement fast route convergence.

#### Pre-configuration Tasks

Before configuring dynamic BFD for IS-IS, complete the following tasks:

#### Configuring BFD Globally

Before configuring dynamic BFD for IS-IS, you need to enable BFD globally.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bfd**

BFD is configured globally.

##### Step 3 Run **commit**

The configuration is committed.

----End

#### Configuring BFD for an IS-IS Process

By configuring BFD for an IS-IS process, you can set parameters for dynamic BFD sessions and enable dynamic BFD for IS-IS on all IS-IS interfaces.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis process-id**

The IS-IS view is displayed.

### Step 3 (Optional) Run **bfd session-up check**

BFD session check is configured for the IS-IS process.



When the Layer 2 network is running normally, IS-IS neighbor relationships can be established and routes can be delivered. However, if the Layer 3 network is unreachable, upper-layer traffic loss occurs. To resolve this problem, configure BFD session check for an IS-IS process using the **bfd session-up check** command. This ensures that an IS-IS neighbor relationship is established only when the BFD session is up on corresponding interfaces. After this function is configured, it applies only to the neighbor relationships to be established, not to existing neighbor relationships.

### Step 4 Run **bfd all-interfaces enable**

BFD is enabled for the IS-IS process to establish BFD sessions.

BFD must be enabled globally before you perform this step. If BFD is enabled globally, this step is performed, and the status of neighbors is up (or the DIS is up in the case of a broadcast network), default BFD parameters are used by all interfaces in the IS-IS process to establish BFD sessions.

### Step 5 (Optional) If you need to set BFD parameters for establishing a BFD session, run the **bfd all-interfaces { min-rx-interval receive-interval | min-tx-interval transmit-interval | detect-multiplier multiplier-value | frr-binding | tos-exp tosexp-value } \* command.**

BFD detection intervals are calculated as follows:

- Effective interval at which BFD packets are sent by the local end = MAX { Configured minimum interval at which BFD packets are sent by the local end, Configured minimum interval at which BFD packets are received by the remote end }
- Effective interval at which BFD packets are received by the local end = MAX { Configured minimum interval at which BFD packets are sent by the remote end, Configured minimum interval at which BFD packets are received by the local end }
- Effective detection interval on the local end = Effective interval at which BFD packets are received by the local end x Detection multiplier configured on the remote end

If **tos-exp tosexp-value** is specified, a priority is configured for all IS-IS BFD packets in the current process.

### Step 6 (Optional) Disable a specified interface from creating a BFD session.

After BFD is configured for an IS-IS process, all interfaces on which neighbor relationships are up in the IS-IS process create BFD sessions. If BFD is not required on specific interfaces, disable these interfaces from creating BFD sessions.

1. Run **quit**  
Return to the system view.
2. Run **interface interface-type interface-number**  
The interface view is displayed.
3. Run **isis bfd block**  
The interface is disabled from creating a BFD session.
4. Run **quit**  
Return to the system view.
5. Run **isis process-id**  
The IS-IS view is displayed.

**Step 7** (Optional) Run **bfd all-interfaces incr-cost { cost-value | max-reachable } [ wtr-value ]**

The IS-IS process is enabled to adjust the cost based on the status of an associated BFD session.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

The delay configured for an interface to restore the cost based on the BFD session status takes precedence over that configured for a process to restore the cost based on the BFD session status.

**Step 8** Run **commit**

The configuration is committed.

----End

## (Optional) Configuring BFD on a Specified Interface

You can configure dynamic BFD session parameters for a specified interface.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** (Optional) To configure BFD session check for an IS-IS process, perform the following steps:

1. Run **isis process-id**  
The IS-IS view is displayed.
2. Run **bfd session-up check**  
BFD session check is configured for the IS-IS process.

 NOTE

When the Layer 2 network is running normally, IS-IS neighbor relationships can be established and routes can be delivered. However, if the Layer 3 network is unreachable, upper-layer traffic loss occurs. To resolve this problem, configure BFD session check for an IS-IS process by running the **bfd session-up check** command. This ensures that an IS-IS neighbor relationship is established only when the BFD session is up on corresponding interfaces. After this function is configured, it applies only to the neighbor relationships to be established (it does not apply to existing neighbor relationships).

**Step 3** Run **interface interface-type interface-number**

The interface view is displayed.

 NOTE

An interface in this case may be a physical interface or a GRE tunnel interface. If BFD is enabled on a GRE tunnel interface, millisecond-level fault detection can be implemented for the GRE tunnel.

**Step 4** Run **isis bfd enable**

BFD is enabled for the interface to establish BFD sessions.

BFD must be enabled globally before you perform this step. If BFD is enabled globally, this step is performed, and the status of neighbors is up (or the DIS is up in the case of a broadcast network), default BFD parameters are used by this interface to establish BFD sessions.

**Step 5** (Optional) Run **isis bfd { min-rx-interval receive-interval | min-tx-interval transmit-interval | detect-multiplier multiplier-value | frr-binding | tos-exp tosexp-value } \***

Parameters used for establishing a BFD session are set.

If **tos-exp tosexp-value** is specified, a priority is configured for all IS-IS BFD packets on the current interface.

 NOTE

The BFD configurations on an interface take precedence over those of a process. That is, if BFD session parameters are configured for both a process and an interface, the parameters on the interface will be used to establish BFD sessions.

**Step 6** (Optional) Run **isis bfd incr-cost { cost-value | max-reachable } [ wtr wtr-value ]**

The interface is enabled to adjust the cost based on the status of an associated BFD session.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

The cost restoration delay configured for BFD association on an interface takes precedence over that configured for BFD association in a process.

**Step 7** (Optional) Run **isis suppress-flapping bfd detecting-interval detecting-interval-value threshold threshold-value incr-cost cost-value wtr wtr-value**

BFD session flapping suppression is enabled on the interface.

### Step 8 Run commit

The configuration is committed.

----End

## Verifying the Configuration of Dynamic BFD for IS-IS

After configuring dynamic BFD for IS-IS, check information about the BFD session and dynamic BFD for IS-IS on an interface.

### Prerequisites

Dynamic BFD for IS-IS has been configured.

### Procedure

- Run the **display isis bfd [ process-id | vpn-instance vpn-instance-name ] session { peer ip-address | all }** command to view the information about the BFD session.
- Run the **display isis bfd [ process-id ] interface** command to view the information about BFD on an interface.

----End

### 1.1.8.2.15 Configuring Track BFD for IS-IS

Track BFD allows you to manually bind an IS-IS interface to a link-bundle BFD session by specifying a session name. This function can quickly detect link faults and prevent BFD session faults caused by faults of the board on which a trunk member interface resides.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run bfd

BFD is enabled.

#### Step 3 Run quit

Return to the system view.

#### Step 4 Run isis [ process-id ]

The IS-IS view is displayed.

#### Step 5 Run network-entity net-addr

A network entity title (NET) is configured.

#### Step 6 Run quit

Return to the system view.

**Step 7** Run **interface interface-type interface-number**

An interface is created.

**Step 8** Run **quit**

Return to the system view.

**Step 9** Run **bfd sess-name bind link-bundle [ compatible-mode ] peer-ip ip-address [ vpn-instance vpn-name ] interface interface-type interface-number source-ip ip-address**

A BFD for link-bundle session is created to detect Eth-Trunk faults, and the BFD session view is displayed.

**Step 10** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 11** Run **isis enable [ process-id ]**

IS-IS is enabled on the interface.

**Step 12** Run **isis bfd track session-name bfd-session-name**

Track BFD is enabled on the interface.

**Step 13** Run **commit**

The configuration is committed.

----End

### 1.1.8.2.16 Configuring IS-IS IPv4 MT to Isolate Multicast Services from Unicast Services

On an IS-IS IPv4 network, IS-IS multi-topology (MT) improves usage of network resources by allowing services to run in multiple logical topologies.

#### Usage Scenario

On a traditional IP network, only one unicast forwarding table is available on the forwarding plane because only one unicast topology exists, which forces services transmitted from one router to the same destination address to share the same next hop, and various end-to-end services such as voice and data services, to share the same physical links. As a result, some links may become heavily congested while others remain relatively idle. In addition, QoS requirements vary based on services. The traditional unicast topology cannot meet various QoS requirements.

IS-IS MT allows multiple separate logical topologies to be deployed in an IS-IS autonomous system (AS). Separate multicast topologies can be set up for multicast services, and this configuration isolates multicast topologies from unicast topologies.

IS-IS MT allows users to configure the network flexibly, reducing network construction cost.

To configure IS-IS MT, perform the following steps. First, associate an IS-IS process with topology instances so that the SPF calculation can be performed for the

topology instances in the IS-IS process. Second, associate a specified interface with the topology instances so that every link can participate in the SPF calculation.

## Pre-configuration Tasks

Before configuring IS-IS MT to isolate IPv4 multicast services from IPv4 unicast services, complete the following tasks:

### Enabling IPv4 MT for an IS-IS Process

Based on service requirements and the network plan, an IS-IS process can be associated with various topology instances so that multiple logical topologies are deployed in an IS-IS AS.

## Context

You can associate an IS-IS process with unicast or multicast topology instances as required so that the SPF calculation can be performed for the topology instances in the IS-IS process. Then, you can configure parameters, such as the interface cost and protocol priority for the topology instances.

The configuration in an IS-IS topology instance view takes effect only on the topology instance. Therefore, you can configure different features and parameters for different topology instances as required.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **ip topology topology-name**

An IPv4 topology instance is configured.

### Step 3 Run **isis [ process-id ]**

The IS-IS view is displayed.

### Step 4 Run **cost-style { wide | wide-compatible }**

The cost style of the packets to be sent and accepted by the router is set to **wide** or **wide-compatible**.

#### NOTE

Go to Step 5 or Step 6 as required.

- If the router supports only unicast services, go to Step 5.
- If the router supports only multicast services, go to Step 6.
- If the router supports both unicast and multicast services, perform the following steps.

### Step 5 Run **topology topology-name topology-id mt-id**

The IS-IS process is associated with a specified unicast topology instance, and the IS-IS unicast topology instance view is displayed.

### Step 6 Run **topology topology-name topology-id multicast**

The IS-IS process is associated with a specified multicast topology instance, and the IS-IS multicast topology instance view is displayed.

**Step 7** (Optional) Configure IS-IS parameters for the IPv4 topology instance. The supported commands are as follows:

- To set the priority of IS-IS, run the **preference** command.
- To enable an IS-IS device to generate a default route, run the **default-route-advertise** command.
- To enable an IS-IS device to import routes from other routing protocols in the IS-IS view, run the **import-route** command.
- To control IS-IS route leaking from a Level-1 area to a Level-2 area, run the **import-route isis level-1 into level-2** command.
- To control IS-IS route leaking from a Level-2 area to a Level-1 area, run the **import-route isis level-2 into level-1** command.
- To enable IS-IS summarization, run the **summary** command.
- To configure a link cost for all interfaces of a specified IS-IS process, run the **circuit-cost { cost | maximum } [ level-1 | level-2 ]** command.
- To enable IS-IS to automatically calculate the interface cost according to the interface bandwidth, run the **auto-cost enable [ compatible ]** command.
- To set a bandwidth reference value used for automatic interface cost calculation, run the **bandwidth-reference value** command.
- To set the maximum number of equal-cost routes for load balancing, run the **maximum load-balancing number** command.

**Step 8** Run **commit**

The configuration is committed.

----End

## Enabling MT for IPv4 an IS-IS Interface

After IS-IS MT is enabled, associate a specified interface with MT instances so that specified links can participate in the SPF calculation for the topology instances.

### Context

Various topology instances can be associated with one interface so that specified links of the interface can participate in the SPF calculation for the topology instances.

The following parameters can be configured for an interface in different topology instances:

- Cost of the interface in each topology instance
- Administrative tag of the interface in each topology instance

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface interface-type interface-number**

The physical interface view is displayed.

**Step 3** Run **ip topology topology-name enable**

An IPv4 topology instance is enabled for the interface.

**Step 4** Run **isis topology topology-name**

The interface is associated with the IS-IS IPv4 topology instance.



If You want to associate an interface with multiple IPv4 topology instances, repeat this step and specify different topology names.

**Step 5** (Optional) Run **isis topology topology-name cost cost [ level-1 | level-2 ]**

A cost is configured for the interface in the IS-IS topology instance.

A cost change on an interface will trigger route reselection. Therefore, change the cost during network planning.

**Step 6** (Optional) Run **isis topology topology-name tag-value tag [ level-1 | level-2 ]**

An administrative tag is configured for the interface in the IS-IS topology instance.

An administrative tag is carried in routes and used by route-policies to filter routes.

**Step 7** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of IS-IS IPv4 Multi-Topologies Used to Isolate Multicast Services from Unicast Services

After configuring IPv4 IS-IS MT to isolate multicast services from unicast services, check the IS-IS MT configurations.

### Prerequisites

IPv4 IS-IS MT has been configured to isolate multicast services from unicast services.

### Procedure

- Run the **display isis peer [ verbose ] [ process-id | vpn-instance vpn-instance-name ]** command to check information about IS-IS neighbors.
- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv4 ] [ topology topology-name ] [ verbose | [ level-1 | level-2 ] | ip-address [ mask | mask-length ] ] \***  command to check IS-IS routing information.

- Run the **display isis spf-tree [ systemid *systemid* ] [ [ level-1 | level-2 ] | topology *topology-name* ] \* verbose [ process-id | vpn-instance *vpn-instance-name* ]** command to check IS-IS SPT information.

----End

### 1.1.8.2.17 Configuring IPv4 IS-IS Route Summarization

To speed up IPv4 route lookup and simplify route management on a large-scale IS-IS network, configure IS-IS route summarization to reduce the number of IS-IS routes in a routing table.

#### Usage Scenario

The IS-IS routing table of a device on a medium or large IS-IS network contains a large number of routing entries. Storing the routing table consumes a large number of memory resources, and transmitting and processing routing information consume lots of network resources. IS-IS route summarization can solve this problem.

Routes with the same IPv4 prefix can be summarized into one route. Route summarization on a large-scale IS-IS network reduces the number of routes in a routing table and minimizes system resource consumption. In addition, if a specific link frequently alternates between Up and Down, the link status changes will not be advertised to devices that are located beyond the summarized route network segment, which prevents route flapping and improves network stability.

#### Pre-configuration Tasks

Before configuring IS-IS route summarization, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring devices are reachable at the network layer.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **isis [ *process-id* ]**

The IS-IS view is displayed.

##### Step 3 Run **summary *ip-address mask* [ avoid-feedback | generate\_null0\_route | tag *tag-value* | [level-1 | level-1-2 | level-2] ] \***

The specified IS-IS routes are summarized into one IS-IS route.



After route summarization is configured on an IS, the local IPv4 routing table still contains all specific routes before the summarization.

The IPv4 routing tables on other ISs that receive the LSP from the local IS contain only the summarized route, and the summarized route is deleted only after all its specific routes are deleted.

#### Step 4 Run commit

The configuration is committed.

----End

### Checking the Configurations

After configuring route summarization, perform the following steps to check whether the route summarization function has taken effect.

- Run the **display isis route** command to check summarized routes in the IS-IS routing table.
- Run the **display ip routing-table [ verbose ]** command to check summarized routes in the IP routing table.

### 1.1.8.2.18 Configuring IS-IS Auto FRR

This section describes how to configure IS-IS Auto FRR.

#### Usage Scenario

As networks develop, Voice over Internet Protocol (VoIP) and online video services pose increasingly higher requirements for real-time transmission. Nevertheless, if an IS-IS fault occurs, multiple operations, including fault detection, LSP updating, LSP flooding, route calculation, and forwarding information base (FIB) entry delivery, must be performed to switch traffic to a new link. These operations take a long time, much longer than 50 ms, the minimum delay to which users are sensitive. As a result, user experience is affected.

With IS-IS Auto FRR, devices can rapidly switch traffic from a faulty link to an alternate one without interrupting the traffic, which significantly improves IS-IS network reliability.

IS-IS Auto FRR is suitable for IP services that are sensitive to delay and packet loss.

#### Pre-configuration Tasks

Before configuring IS-IS Auto FRR, complete the following tasks:

- Configure IP addresses for interfaces and ensure that neighboring devices are reachable at the network layer.
- **Configure basic IPv4 IS-IS functions.**
- Configure a local LDP session on the source node, the PQ node, and the nodes between them if remote LFA FRR is required.

#### Procedure

##### Step 1 Run system-view

The system view is displayed.

##### Step 2 Run isis [ process-id ]

An IS-IS process is created, and the IS-IS view is displayed.

### Step 3 Run frr

The IS-IS FRR view is displayed.

### Step 4 (Optional) Run **ecmp disable { level-1 | level-2 }**

IS-IS ECMP FRR is disabled.

With ECMP FRR, IS-IS pre-calculates backup paths for load balancing links based on the LSDBs on the entire network. The backup paths are stored in the forwarding table and are used for traffic protection in the case of link failures. If the network topology changes, ECMP FRR triggers calculation of new backup paths, which increases the system calculation pressure. In addition, the backup entries generated by the ECMP FRR function increase the memory usage of the system. In this case, you can run the **ecmp disable** command to disable the ECMP FRR function.

### Step 5 Run **loop-free-alternate [ level-1 | level-2 | level-1-2 ]**

IS-IS Auto FRR is enabled to generate a loop-free backup link.

If no level is specified, IS-IS Auto FRR is enabled for Level-1 and Level-2 to generate backup routes.

#### NOTE

IS-IS can create loop-free backup links only if the traffic protection inequality of IS-IS Auto FRR is met.

### Step 6 (Optional) Run **frr-policy route route-policy route-policy-name**

A route-policy is configured to filter IS-IS alternate routes.

### Step 7 If you want to configure remote LFA Auto FRR, perform the following steps:

1. Run **remote-lfa tunnel ldp [ maximum-reachable-cost cost-value ] [ level-1 | level-2 | level-1-2 ]**

Remote LFA Auto FRR is enabled.

2. (Optional) Run **remote-lfa available-tunnel-destination ip-prefix ip-prefix-name [ level-1 | level-2 | level-1-2 ]**

A filtering policy is configured to filter PQ nodes.

Only the PQ node that matches the filtering policy can be used as the next hop of an LFA link.

3. Run **quit**

Return to the IS-IS view.

4. (Optional) Run **avoid-microloop frr-protected**

The IS-IS anti-microloop function is enabled.

5. (Optional) Run **avoid-microloop frr-protected rib-update-delay rib-update-delay**

The delay after which IS-IS delivers routes is configured.

If IS-IS remote LFA FRR is enabled and the primary link fails, traffic is switched to the backup link. If route convergence occurs again, traffic is

switched from the backup link to a new primary link. During the switchover, microloop may occur. To prevent this problem, IS-IS anti-microloop is enabled can be used to delay the switching. To configure the delay, run the **avoid-microloop frr-protected rib-update-delay** *rib-update-delay* command. After the command is run, IS-IS does not switch traffic to the backup link until the delay expires.

Keeping the default delay is recommended.

6. Run **frr**

The IS-IS FRR view is displayed.

**Step 8** (Optional) Run **tiebreaker { node-protecting | lowest-cost | non-ecmp | srlg-disjoint | hold-max-cost } preference** *preference* [ **level-1 | level-2 | level-1-2** ]

The solution of selecting a backup path for IS-IS Auto FRR is set.

**Step 9** (Optional) To prevent an interface from being calculated as a backup interface (after the preceding configurations are complete, all IS-IS interfaces participate in the calculation of the backup next hop by default), perform the following steps:

1. Run **quit**

Exit the IS-IS FRR view.

2. Run **quit**

Exit the IS-IS view.

3. Run **interface** *interface-type interface-number*

The interface view is displayed.

4. Run **undo isis lfa-backup** [ **level-1 | level-2 | level-1-2** ]

The interface is disabled from being calculated as an LFA or remote LFA next hop.

**Step 10** (Optional) Run **isis remote-lfa disable** [ **level-1 | level-2 | level-1-2** ]

The system is prevented from calculating a remote LFA next hop for the IS-IS route whose outbound interface is the current interface.

**Step 11** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring IS-IS Auto FRR, check the configurations.

- Run the **display isis route** [ **level-1 | level-2** ] [ *process-id* ] [ **verbose** ] command to check information about the primary and alternate links after IS-IS Auto FRR is enabled.
- Run the **display isis spf-tree** [ [ **level-1 | level-2** ] | **verbose** ] \* [ *process-id* | **vpn-instance** *vpn-instance-name* ] command to check the traffic protection type of IS-IS Auto FRR.
- Run the **display isis frr summary** [ **level-1 | level-2 | level-1-2** ] [ *process-id* ] command to view the FRR protection rates of routes in an IS-IS process.

- Run the **display isis [ process-id ] srlg { srlgGroupId | all }** command to check shared risk link group (SRLG) information.

### 1.1.8.2.19 Configuring Basic IPv6 IS-IS Functions

This section describes how to configure basic IPv6 IS-IS functions for communication between nodes on an IPv6 IS-IS network. The configuration procedure includes configuring the IS-IS process and IS-IS interface.

#### Usage Scenario

To deploy IS-IS on an IPv6 network, configure basic IS-IS functions to implement communication between different nodes on the network.

Other IS-IS functions can be configured only after basic IS-IS functions are configured.

Configuring basic IPv6 IS-IS functions includes the following operations:

- Create IPv6 IS-IS processes.
- Configure IPv6 IS-IS interfaces.

#### Pre-configuration Tasks

Before configuring basic IPv6 IS-IS functions, complete the following tasks:

- Configure a link layer protocol.
- Assign an IPv6 address to each interface to ensure IP connectivity.
- Enable the IPv6 in system view.

#### Creating an IPv6 IS-IS Process

Before configuring basic IPv6 IS-IS features, create an IPv6 IS-IS process and then enable IPv6 IS-IS interfaces.

#### Context

To create an IPv6 IS-IS process, perform the following operations:

- Create an IS-IS process and configure the NET of a device.**
- (Optional) Configure the level of a device.**

Configure the device level based on the network planning. If no device level is configured, IS-IS establishes separate neighbor relationships for Level-1 and Level-2 devices and maintains two identical LSDBs, consuming excessive system resources.

- (Optional) Configure IS-IS host name mapping.**

After IS-IS host name mapping is configured, a host name rather than the system ID of a device is displayed when you run a display command. This configuration improves the maintainability on an IS-IS network.

- (Optional) Enable IS-IS adjacency strict-check.**

If both IPv4 and IPv6 run on a network, and the IPv6 topology type of this network is **standard** or **compatible**, enable IS-IS adjacency strict-check to ensure that an IS-IS adjacency is established only when both IPv4 and IPv6 go

Up. IS-IS adjacency strict-check improves network reliability and prevents traffic losses.

## Procedure

- Create an IS-IS process, configure the NET of a device, and enable IPv6 for the process.

- a. Run **system-view**

The system view is displayed.

- b. Run **isis [ process-id ]**

An IS-IS process is created, and the IS-IS process view is displayed.

The *process-id* parameter specifies the ID of an IS-IS process. The default value of *process-id* is 1. To associate an IS-IS process with a VPN instance, run the **isis process-id vpn-instance vpn-instance-name** command.

- c. Run **network-entity net-addr**

A NET is configured.

NET of IS-IS consists of three parts:

- Part one is the area ID that is variable (1 to 13 bytes), and the area IDs of the devices in the same area are identical.
- Part two is the system ID (6 bytes) of this device, which must be unique in the whole area and backbone area.
- Part three is the last byte "SEL", whose value must be "00".

For example, the NET of IS-IS device can be configured as 10.1234.6e9f.0001.00.

---

### NOTICE

- An area ID is used to uniquely identify an area in the same IS-IS domain. All routers in the same Level-1 area must share the same area ID, while routers in the same Level-2 area can have different area IDs.
- The system ID must be unique in the whole area and backbone area.
- When configuring multiple NETs, ensure that they share the same system ID.

Configuring loopback interface addresses based on NETs is recommended to ensure that a NET is unique on the network. If NETs are not unique, route flapping will easily occur. System ID used in IS-IS can be obtained in the following way: extend each part of the IP address to 3 bits, add 0 to the front of any part that is shorter than 3 bits, divide the extended address into three parts, with each part consisting of four decimal digits, and the reconstructed address is the system ID.

- d. Run **ipv6 enable [ topology { compatible [ enable-mt-spf ] | ipv6 | standard } ]**

- The IPv6 of IS-IS process is enabled.
- e. Run **commit**
  - The configuration is committed.
- (Optional) Configure the level of a device.
  - a. Run **is-level { level-1 | level-1-2 | level-2 }**
    - The level of the router is configured.
  - b. Run **commit**
    - The configuration is committed.
- (Optional) Configure IS-IS host name mapping.
  - a. Run **is-name *symbolic-name***
    - IS-IS dynamic host name mapping is configured. The system ID of the local device is mapped to the specified host name.
    - The value of *symbolic-name* is contained in LSP packets and advertised to other IS-IS devices.
    - symbolic-name* rather than the system ID of the local IS-IS device is displayed on another IS-IS device.
  - b. Run **is-name map *system-id symbolic-name***
    - IS-IS static host name mapping is configured. The system ID of a peer IS-IS device is mapped to the specified host name.
    - This command configuration takes effect only on the local IS-IS device.
    - The value of *symbolic-name* will not be added to LSP packets.
- c. Run **commit**
  - The configuration is committed.
- (Optional) Enable IS-IS adjacency strict-check.
  - a. Run **adjacency-strict-check enable**
    - IS-IS adjacency strict-check is enabled.
  - b. Run **commit**
    - The configuration is committed.

----End

## Configuring an IPv6 IS-IS Interface

To configure an interface on an IS-IS device to send Hello packets or flood LSPs, IS-IS must be enabled on this interface.

## Context

The level of an IS-IS device and level of an interface determine the level of a neighbor relationship. By default, Level-1 and Level-2 neighbor relationships will be established between two Level-1-2 devices. If only one level of neighbor

relationships is required, you can configure the level of an interface to prevent the establishment of neighbor relationships of the other level.

After IS-IS is enabled on an interface, the interface will automatically send Hello packets, attempting to establish neighbor relationships. If a peer device is not an IS-IS device or if an interface is not expected to send Hello packets, suppress the interface. Then this interface only advertises routes of the network segment where the interface resides but does not send Hello packets. This suppression reduces the link bandwidth usage.

## Procedure

- Configure an IS-IS interface.

- Run **system-view**

The system view is displayed.

- (Optional) Run **isis interface limit disable**

The limit on the number of IS-IS interfaces that can be configured on the device is disabled.

After this command is run, the number of IS-IS interfaces is no longer limited to the device threshold. To restore the limit on the device, run the **undo isis interface limit disable** command. If the number of IS-IS interfaces on the device is greater than or equal to the threshold before the configuration, no more IS-IS interfaces can be added on the device.

- Run **interface interface-type interface-number**

The interface view is displayed.

- Run **ipv6 enable**

IPv6 is enabled on the interface.

- Run **isis ipv6 enable [ process-id ]**

IS-IS IPv6 is enabled for the interface.

After this command is run, the IS-IS device uses the specified interface to send Hello packets and flood LSPs.

 **NOTE**

No neighbor relationship needs to be established between loopback interfaces. Therefore, if this command is run on a loopback interface, the routes of the network segment where the loopback interface resides will be advertised through other IS-IS interfaces.

- Run **commit**

The configuration is committed.

- (Optional) Configure the level of an IS-IS interface.

- Run **isis circuit-level [ level-1 | level-1-2 | level-2 ]**

The level of the interface is configured.

 NOTE

Changing the level of an IS-IS interface is valid only when the level of the IS-IS device is Level-1-2. If the level of the IS-IS device is not a Level-1-2, the level of the IS-IS device determines the level of the adjacency to be established.

b. Run **commit**

The configuration is committed.

- (Optional) Suppress an IS-IS interface.

a. Run **isis silent [ advertise-zero-cost ]**

The IS-IS interface is suppressed.

b. Run **commit**

The configuration is committed.

----End

## (Optional) Configuring a Cost for IS-IS Interfaces (IPv6)

Configuring IS-IS interface costs can control IS-IS route selection. Set interface costs based on network planning.

### Context

The costs of IS-IS interfaces can be determined in the following modes (in descending order of priority):

- The interface cost takes effect only on a specified interface.
- The global cost takes effect only on all interfaces.
- The automatically calculated cost is a cost automatically calculated based on the interface bandwidth.

The default cost of an IPv6 IS-IS interface is 10, and the default cost style is narrow.

### Procedure

- Configure an IS-IS cost style.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run **cost-style { narrow | wide | wide-compatible | { narrow-compatible | compatible } [ relax-spf-limit ] }**

An IS-IS cost style is configured.

The cost value range of an interface and the cost value range of the routes that can be accepted by the interface vary with the cost style.

- If the cost style is **narrow**, the cost of an interface ranges from 1 to 63. The maximum cost of the routes accepted by the device is 1023.

- If the cost style is **narrow-compatible** or **compatible**, the cost of an interface ranges from 1 to 63. The cost of the routes accepted by the device is related to **relax-spf-limit**.
  - If **relax-spf-limit** is not specified:
    - If the cost of a route is less than or equal to 1023 and the cost of each interface through which the route passes is less than or equal to 63, the cost of the route received by the interface is the actual cost.
    - If the cost of a route is less than or equal to 1023 but the link costs of some interfaces through which the route passes are greater than 63: An IS-IS device can learn only the routes to the network segment where the interface resides and the routes imported by the interface. The cost of the route received by the interface is the actual one, and subsequent routes forwarded to the interface are discarded.
    - If the cost of a route is greater than 1023, the device can learn only the routes of the interface whose link cost exceeds 1023 for the first time (all the original link costs of the interface are not greater than 63). The routes to the network segment where the interface resides and the routes imported by the interface can be learned. The cost of the routes is 1023. Subsequent routes forwarded to the interface are discarded.
  - If **relax-spf-limit** is set:
    - There is no limit on link costs of interfaces or route costs. The local device considers the cost of each route as the actual one and accepts the routes.
- If the cost style is **wide** or **wide-compatible**, the cost of an interface ranges from 1 to 16777214 or the maximum value (16777215). If **maximum** is configured, the neighbor TLV with cost 16777215 generated on the link is used to transmit TE information, not for route calculation. The maximum cost of the routes accepted by the device is 0xFFFFFFFF.

d. Run **commit**

The configuration is committed.

- Configure a cost for the IS-IS interface.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis ipv6 [ topology topology-name ] cost cost-value [ level-1 | level-2 ]**

A cost is configured for the IS-IS interface.

d. Run **commit**

The configuration is committed.

- Configure the global IS-IS cost.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Run **ipv6 circuit-cost { cost } [ level-1 | level-2 ]**

A global IS-IS cost is configured.

You can run this command to change the costs of all interfaces at a time.
  - d. Run **commit**

The configuration is committed.
- Enable IS-IS to automatically calculate interface costs.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Run **ipv6 bandwidth-reference value**

The reference value of the bandwidth is configured.
  - d. Run **ipv6 auto-cost enable [ compatible ]**

IS-IS is enabled to automatically calculate the interface cost.

- If the cost style of the system is wide or wide-compatible, when **ipv6 auto-cost enable** command is configured, Interface cost = (Bandwidth-reference/Interface bandwidth) x 10, and when **ipv6 auto-cost enable compatible** command is configured, Interface cost = (Bandwidth-reference/Interface bandwidth).

 NOTE

If the interface cost calculated through this formula is greater than 16777214, 16777214 is used as the interface cost for route calculation. That is, the interface cost will never be greater than 16777214.

The **ipv6 auto-cost enable** command can be run on Eth-Trunk interfaces as same with on physical interfaces. If the command is run on an Eth-Trunk interface, the bandwidth of the Eth-Trunk interface is equal to the total bandwidth of all its member interfaces.

- If the cost-style is narrow, narrow-compatible, or compatible, the cost of each interface is determined by the interface bandwidth range. **Table 1-92** lists the interface costs corresponding to different interface bandwidth ranges.

**Table 1-92** Mapping between IS-IS interface costs and interface bandwidth

| Cost | Bandwidth Range                                    |
|------|----------------------------------------------------|
| 60   | Interface bandwidth $\leq$ 10 Mbit/s               |
| 50   | 10 Mbit/s < interface bandwidth $\leq$ 100 Mbit/s  |
| 40   | 100 Mbit/s < interface bandwidth $\leq$ 155 Mbit/s |
| 30   | 155 Mbit/s < interface bandwidth $\leq$ 622 Mbit/s |
| 20   | 622 Mbit/s < Interface bandwidth $\leq$ 2.5 Gbit/s |
| 10   | Interface bandwidth $>$ 2.5 Gbit/s                 |

 **NOTE**

To change the cost of a loopback interface, run the **isis ipv6 cost** command only in the loopback interface view.

e. Run **commit**

The configuration is committed.

----End

## (Optional) Configuring IPv6 IS-IS Attributes for Interfaces on Networks of Different Types

Different IS-IS attributes can be configured for different types of network interfaces.

### Context

The establishment mode of IS-IS neighbor relationships on a broadcast network is different from that on a P2P network. Different IS-IS attributes can be configured for interfaces on different types of networks.

IS-IS is required to select a Designated Intermediate System (DIS) on a broadcast network. Configure the DIS priorities of IS-IS interfaces so that the interface with the highest priority is selected as the DIS.

The network types of the IS-IS interfaces on both ends of a link must be the same, otherwise, the IS-IS neighbor relationship cannot be established between the two interfaces. If the type of an interface on the neighbor is P2P, you can configure the interface type on the local device to P2P so that an IS-IS neighbor relationship can be established between the two devices.

IS-IS on a P2P network is not required to select a DIS. Therefore, you do not need to configure DIS priorities. To ensure the reliability of P2P links, configure IS-IS to use the three-way handshake mode for IS-IS neighbor relationship establishment so that faults on a unidirectional link can be detected.

## Procedure

- Configure the DIS priority of an IS-IS interface.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **interface interface-type interface-number**

The interface view is displayed.
  - c. Run **isis dis-priority priority [ level-1 | level-2 ]**

The DIS priority is configured on the interface. The greater the value, the higher the priority.
  - d. Run **commit**

The configuration is committed.
- Configure the network type of an IS-IS interface.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **interface interface-type interface-number**

The interface view is displayed.
  - c. Run **isis circuit-type p2p**

The network type of the interface is set to P2P.

When the network type of an IS-IS interface changes, interface configurations change accordingly.
    - After a broadcast interface is configured as a P2P interface using the **isis circuit-type p2p** command, the default values of the interval at which Hello packets are sent, the number of Hello packets that IS-IS fails to receive from a neighbor before declaring the neighbor Down, the interval LSPs are retransmitted on a P2P link, and various IS-IS authentication modes take effect. Consequently, other configurations such as the DIS priority, DIS name, and interval at which CSNPs are sent on a broadcast network become invalid.
    - After the **undo isis circuit-type** command is run to restore the network type, the default values of the interval at which Hello packets are sent, the number of Hello packets that IS-IS fails to receive from a neighbor before declaring the neighbor Down, the interval LSPs are retransmitted on a P2P link, the IS-IS authentication mode, the DIS priority, and the interval at which CSNPs are sent on a broadcast network take effect.
  - d. Run **commit**

The configuration is committed.
- Set the negotiation mode for the establishment of neighbor relationships over P2P links.
  - a. Run **system-view**

The system view is displayed.

- b. Run **interface interface-type interface-number**  
The interface view is displayed.
  - c. Run **isis ppp-negotiation { 2-way | 3-way [ only ] }**  
The negotiation mode is specified on the interface.  
This command applies only to neighbor relationship establishment over P2P links. In the case of a broadcast link, you can run the **isis circuit-type p2p** command to set the link type to P2P, and then run the **isis ppp-negotiation** command to set the negotiation mode for the establishment of the neighbor relationship.
  - d. Run **commit**  
The configuration is committed.
- Configure OSICP negotiation check on PPP interfaces.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **interface interface-type interface-number**  
The interface view is displayed.
    - c. Run **isis ppp-osicp-check**  
PPP OSICP status check is enabled.  
This command is applicable only to PPP interfaces. For P2P interfaces, this command is invalid.  
After this command is run, OSI network negotiation status of PPP affects IS-IS interface status. When PPP detects that the OSI network fails, the link status of the IS-IS interface goes down and the route to the network segment where the interface resides is not advertised through LSPs.
    - d. Run **commit**  
The configuration is committed.

----End

## (Optional) Configuring IS-IS to Adjust the Flooding Rate (IPv6)

### Context

In a scenario where a large number of LSPs need to be flooded at a time, for example, in a large-scale network, because the maximum number of LSPs that can be sent by IS-IS per second is limited, the time needed to complete the LSP flooding at a time may exceed the expected time, which affects the convergence efficiency on the entire network. In this case, you can increase the maximum IS-IS LSP flooding rate by increasing the maximum number of LSPs that can be sent per second during IS-IS LSP flooding. This speeds up IS-IS LSP flooding and network convergence. When the flooding pressure on the network is heavy and flow control is required, you can reduce the maximum number of LSPs that can be sent by IS-IS per second to slow down the IS-IS LSP flooding rate and relieve the flooding pressure on nodes.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **isis lsp flood-control max-count *max-count-value***

The maximum number of LSPs that IS-IS sends every second is set.

### Step 3 Run **commit**

The configuration is committed.

----End

## (Optional) Enabling LSP fragment extension on an IPv6 IS-IS device

If the LSP capacity is insufficient, newly imported routes and new TLVs fail to be added to LSP fragments. In this case, you can use LSP fragment extension to increase the LSP capacity, restoring the LSP space. When the LSP space is restored, the system automatically attempts to re-add these routes and TLVs to LSP fragments.

## Context

The **lsp-fragments-extend** command enables LSP fragment extension on an IS-IS device in a specified mode and at a specified level. One LSP fragment occupies only one byte and therefore a maximum of 256 fragments are supported. If there are a great number of LSPs and the number of fragments exceeds 256, some information is lost. LSP fragment extension is introduced to address such a problem. You can run the **virtual-system** command to configure one or more virtual systems to support more than 256 LSP fragments.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **isis [ *process-id* ]**

An IS-IS process is created, and the IS-IS process view is displayed.

### Step 3 Run **lsp-fragments-extend [ [ **level-1** | **level-2** | **level-1-2** ] | [ **mode-1** | **mode-2** ] ]<sup>\*</sup>**

Enable LSP fragment extension on an IS-IS device in a specified mode and at a specified level.

----End

## Verifying the Basic IPv6 IS-IS Configuration

After configuring basic IPv6 IS-IS features, check information about IS-IS neighbors, interfaces, and routes.

## Prerequisites

Basic IPv6 IS-IS functions have been configured.

## Procedure

- Step 1** Run the **display isis name-table** [*process-id* | **vpn-instance** *vpn-instance-name*] command to check the mapping between the hostname of the local device and the system ID.
- Step 2** Run the **display isis peer** [**verbose**] [*process-id* | **vpn-instance** *vpn-instance-name*] command to check information about IS-IS neighbors.
- Step 3** Run the **display isis interface** [ [ **verbose** | **traffic-eng** ] \* | **te-tunnel** ] [*process-id* | **vpn-instance** *vpn-instance-name*] command to check information about IS-IS interfaces.
- Step 4** Run the **display isis route** [ *process-id* | **vpn-instance** *vpn-instance-name* ] **ipv6** [ **topology** *topology-name* ] [ **verbose** | [ **level-1** | **level-2** ] | **ipv6-address** [ **prefix-length** ] ] \* [ | **count** ] command to check information about IS-IS routes.

----End

### 1.1.8.2.20 Configuring IPv6 IS-IS Route Selection

Configuring IS-IS route selection can achieve refined control over route selection.

## Usage Scenario

After basic IPv6 IS-IS functions are configured, IS-IS routes will be generated, enabling communication between different nodes on a network.

If multiple routes are available, the route discovered by IS-IS may not be the expected one, which does not meet network planning requirements nor facilitate traffic management. To address this issue, configure IPv6 IS-IS route selection to implement refined control over route selection.

To implement refined control over IPv6 IS-IS route selection, perform the following operations:

- **Configure the costs for IPv6 IS-IS Interfaces**



Changing the IS-IS cost for an interface can control route selection, but routes on the interface need to be recalculated if a network topology changes, especially on a large-scale network. In addition, the configuration result may not meet your expectation.

Therefore, configure IS-IS costs before configuring basic IS-IS functions.

- Configure IPv6 IS-IS route leaking.
- Configure rules for selecting equal-cost IPv6 IS-IS routes.
- Filter IPv6 IS-IS routes.
- Configure an overload bit for an IPv6 IS-IS device.

## Pre-configuration Tasks

Before configuring IPv6 IS-IS route selection, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic IPv6 IS-IS functions.**

## Configuring IPv6 IS-IS Route Leaking

Configuring IS-IS route leaking enables you to optimize IS-IS route selection on a two-level-area network.

### Context

If multiple Level-1-2 devices in a Level-1 area are connected to devices in the Level-2 area, a Level-1 LSP sent by each Level-1-2 device carries an ATT flag bit of 1. This Level-1 area will have multiple routes to the Level-2 area and other Level-1 areas.

By default, routes in a Level-1 area can leak to the Level-2 area so that Level-1-2 and Level-2 devices can learn about the topology of the entire network. Devices in a Level-1 area are unaware of the entire network topology because they only maintain LSDBs in the local Level-1 area. Therefore, a device in a Level-1 area can forward traffic to a Level-2 device only through the nearest Level-1-2 device. However, the used route may not be optimal.

To help a device in a Level-1 area to select the optimal route to another area, configure IPv6 IS-IS route leaking so that some routes in the Level-2 area can leak to the local Level-1 area.

If you want the Level-2 area to know only some of the routes in the local Level-1 area, configure a policy so that only desired routes can leak to the Level-2 area.

### Procedure

- Configure route leaking from the Level-2 area to the Level-1 area.
  - Run **system-view**  
The system view is displayed.
  - Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - Configure the routes in the Level-2 area and other Level-1 areas that meet the specified conditions to leak to the local Level-1 area.

Run any of the following commands as required:

- Configure a basic ACL:
  - Run the **ipv6 import-route isis level-2 into level-1 [ filter-policy { acl6-number| acl6-name acl6-name-string } | tag tag | no-bier ] \* command.**
  - Run **quit**  
Return to the system view.
  - Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

Run **rule** [ *rule-id* ] [ **name** *rule-name* ] { **deny** | **permit** }  
[ **fragment** | **source** { *source-ipv6-address* { *prefix-length* |  
*source-wildcard* } | *source-ipv6-address/prefix-length* | **any** } |  
**time-range** *time-name* | [ **vpn-instance** *vpn-instance-name* |  
**vpn-instance-any** ] ] \*

A rule is configured for the ACL.

- 4) When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- o If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- o If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- o If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- o If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- o Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- Configure an IP prefix list:

Run the **ipv6 import-route isis level-2 into level-1** [ **filter-policy** *ipv6-prefix* *ipv6-prefix-name* | **tag** *tag* | **no-bier** ] \* command.

- Configure a route policy:

Run the **ipv6 import-route isis level-2 into level-1** [ **filter-policy** **route-policy** *route-policy-name* | **tag** *tag* | **no-bier** ] \* command.

#### NOTE

The command is run on the Level-1-2 device that is connected to an external area.

d. Run **commit**

The configuration is committed.

- Configure route leaking from the Level-1 area to the Level-2 area.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

- Configure the routes that meet the specified conditions in the Level-1 area to leak to the Level-2 area.

Run any of the following commands as required:

- Configure a basic ACL:

1) Run the **ipv6 import-route isis level-1 into level-2 [ filter-policy { acl6-number | acl6-name acl6-name-string } | tag tag | no-bier ] \* command.**

2) Run **quit**

Return to the system view.

3) Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

4) Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \***

A rule is configured for the ACL.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the

route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- Configure an IP prefix list:

Run the **ipv6 import-route isis level-1 into level-2 [ filter-policy ipv6-prefix ipv6-prefix-name | tag tag | no-bier ] \* command.**

- Configure a route policy:

Run the **ipv6 import-route isis level-1 into level-2 [ filter-policy route-policy route-policy-name | tag tag | no-bier ] \* command.**

#### NOTE

The command is run on the Level-1-2 device that is connected to an external area.

#### d. Run **commit**

The configuration is committed.

----End

## Configuring a Method for IS-IS to Process Equal-Cost Routes (IPv6)

If multiple equal-cost IPv6 routes are available on an IS-IS network, you can configure load balancing to increase the bandwidth usage of each link, or configure weights for the equal-cost routes to facilitate service traffic management.

## Context

When there are multiple redundant IPv6 links on an IPv6 IS-IS network, multiple equal-cost IPv6 routes may exist. You can configure a method for IPv6 IS-IS to process these equal-cost routes as required:

- Configure load balancing so that traffic is balanced among links.

This method improves link utilization and reduces the possibility of congestion caused by link overload. However, load balancing randomly forwards traffic, which may affect service traffic management.

- Configure weights for equal-cost routes so that the route with the highest priority is preferentially selected, with others functioning as backups.

In this mode, you can allow one or more routes to be preferentially selected without modifying the original configuration. This ensures network reliability and facilitates service traffic management.

## Procedure

- Configure IPv6 IS-IS load balancing.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Run **ipv6 enable**

IPv6 is enabled for the IS-IS process.
  - d. Run **ipv6 maximum load-balancing number**

The maximum number of equal-cost routes for load balancing is set.

### NOTE

If the equal-cost routes on the network outnumber the value specified in the **ipv6 maximum load-balancing** command, routes are selected for load balancing based on the following rules:

1. Route weight: Routes with small weight values (high priority) are selected for load balancing.
  2. Next-hop system ID: If routes have the same weight, those with small system IDs are selected for load balancing.
  3. Outbound interface index: If routes have the same weight and system ID, those with small outbound interface indexes are selected for load balancing.
  4. Next-hop IP address: If the weights, next-hop system IDs, and local outbound interface indexes of the routes are the same, their next-hop IP addresses are compared. The routes with high IP addresses are selected for load balancing.
- e. Run **commit**

The configuration is committed.
  - Configure weights for equal-cost IPv6 IS-IS routes.
    - a. Run **system-view**

The system view is displayed.
    - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
    - c. Run **ipv6 enable**

IPv6 is enabled for the IS-IS process.
    - d. Run **ipv6 nexthop ip-address weight weight-value**

A weight is configured for an equal-cost route. A smaller *weight-value* indicates a higher priority.
    - e. Run **commit**

The configuration is committed.

----End

## Controlling Whether to Add IS-IS Routes to the IPv6 Routing Table

If you do not want some IS-IS routes to be selected, you can configure a policy to prevent these routes from being added to the IP routing table.

### Context

IP packets are forwarded based on the IP routing table. Routes in the IS-IS routing table take effect only after they are successfully added to the IP routing table.

Therefore, you can configure a basic ACL, IPv6 prefix list, or route-policy to allow only the matched IS-IS routes to be delivered to the IP routing table. Unmatched IS-IS routes are neither added to the IP routing table nor selected.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

#### Step 3 Configure a policy to add only some IS-IS routes to the IP routing table.

Run any of the following commands as required:

- Configure a basic ACL:

- a. Run the **ipv6 filter-policy { acl6-number | acl6-name acl6-name } import** command.
- b. Run **quit**  
Return to the system view.
- c. Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**  
The ACL view is displayed.
- d. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ] \***  
A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.

- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:
  - If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.  
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.
- Based on an IP prefix list:  
Run the **ipv6 filter-policy ipv6-prefix *ipv6-prefix-name* import** command.
- Based on a route-policy:  
Run the **ipv6 filter-policy route-policy *route-policy-name* import** command.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring an Overload Bit for an IPv6 IS-IS Device

If an IS-IS device needs to be temporarily isolated, configure the overload state for it to prevent other devices from forwarding traffic to this IS-IS device and prevent routing black holes.

### Context

If an IS-IS device needs to be temporarily isolated, for upgrade or maintenance purposes for example, configure the overload state for it so that no other devices will forward traffic to this IS-IS device.

IS-IS routes converge more quickly than BGP routes do. To prevent routing black holes on a network where both IS-IS and BGP are configured, set an overload bit to instruct an IS-IS device to enter the overload state during its start or restart. After BGP convergence is complete, cancel the overload bit.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

### Step 3 Run **set-overload { on-startup [ timeout1 | start-from-nbr system-id [ timeout1 [ timeout2 ] ] | wait-for-bgp [ timeout1 ] ] [ route-delay-distribute timeout4 ] [ send-sa-bit [ timeout3 ] ] [ route-max-metric ] } [ allow { interlevel | external } \* ]**

The overload bit is configured.

### Step 4 Run **commit**

The configuration is committed.

----End

## Configuring IS-IS to Generate IPv6 Default Routes

This section describes how to configure IS-IS to generate IPv6 default routes to control the advertising of IS-IS routing information.

## Context

The IPv6 default route is the route ::/0. If the destination address of a packet does not match any entry in the routing table of a device, the device sends the packet along the default route. If neither the default route nor the destination address of the packet exists in the routing table, the device discards the packet and informs the source end that the destination address or network is unreachable.

IS-IS can generate default routes using either of the following mode:

- **Command-triggered default route generation**

After the device is configured to advertise the default route, the device adds the default route to an LSP before sending the LSP to a neighbor. In this way, the neighbor can learn this default route.

- **ATT bit setting-triggered default route generation**

As defined in IS-IS, a Level-1-2 router sets the ATT bit in the Level-1 LSPs to be advertised if the Level-1-2 router can reach more Level-1 areas through the Level-2 area than through the Level-1 area. After a Level-1 router receives the LSP with the ATT bit set, it generates a default route destined for the Level-1-2 router that sent this LSP. Based on network requirements, you can configure whether the ATT bit is set and whether a Level-1 router generates a default route after it receives an LSP in which the ATT bit is set.



This mode applies only to Level-1 routers.

## Procedure

- Configure command-triggered default route generation mode.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **ipv6 default-route-advertise [ always | match default | route-policy route-policy-name | route-filter route-filter-name ] [ [ cost cost ] | [ tag tag ] | [ level-1 | level-2 | level-1-2 ] ] \* [ avoid-learning | learning-avoid-loop ]**  
IS-IS is configured to generate default IPv6 routes.
  - d. Run **commit**  
The configuration is committed.
- Configure ATT bit setting-triggered default route generation mode.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Perform the following steps as required:
    - To set the ATT bit in the LSPs sent by the Level-1-2 router, run the **attached-bit advertise { always | never }** command.
      - If the **always** parameter is specified, the ATT bit is always set. After receiving the LSPs in which the ATT bit is set, the Level-1 router generates a default route.
      - If the **never** parameter is specified, the ATT bit is never set. After receiving the LSPs carrying the ATT bit that is not set, each Level-1 router does not generate a default route, which reduces the size of its routing table.
    - To disable the Level-1 router from generating default routes even though it receives the Level-1 LSPs in which the ATT bit is set, run the **attached-bit avoid-learning** command.
  - d. Run **commit**  
The configuration is committed.

----End

## Configuring an IPv6 IS-IS Interface to Automatically Adjust the Link Cost

Configuring an IPv6 IS-IS interface to automatically adjust the link cost based on link quality facilitates route selection control and improves network reliability.

## Context

A bit error refers to the deviation between a bit that is sent and the bit that is received. The bit error rate (BER) refers to the number of bit errors divided by the

total number of bits transferred during a studied time interval. During data transmission, a high BER may degrade or even interrupt services.

To prevent this problem, configure IPv6 IS-IS interfaces to automatically adjust link costs based on link quality so that unreliable links are not used by the optimal routes.

## Procedure

- Perform the following steps for a conventional IS-IS process:
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **ipv6 enable topology ipv6**  
IPv6 is enabled for the IS-IS process.
  - d. Run **quit**  
Return to the system view.
  - e. Run **interface interface-type interface-number**  
The interface view is displayed.
  - f. Run **ipv6 enable**  
IPv6 is enabled on the interface.
  - g. Run **isis ipv6 enable [ process-id ]**  
IPv6 IS-IS is enabled on the interface, and the specified process ID is associated with the interface.
  - h. Run **isis ipv6 link-quality low incr-cost { cost-value | max-reachable }**  
The IPv6 IS-IS interface is enabled to automatically adjust the link cost based on link quality.

### NOTE

The *cost* parameter specifies the link cost adjustment value. After this parameter is specified:

- If the link quality changes from good to low, the link cost equals the original link cost plus the adjustment value. If the new link cost exceeds the maximum link cost allowed, the maximum link cost allowed applies:

- The maximum link cost is 63, if the cost style is **narrow**, **narrow-compatible**, or **compatible**.
  - The maximum link cost is 16777214, if the cost style is **wide** or **wide-compatible**.
  - If the link quality changes from low to good, the original link cost applies.

- i. Run **commit**

The configuration is committed.

- Perform the following steps for an IS-IS multi-instance process:

- a. Run **system-view**

- The system view is displayed.
- b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
- c. Run **ipv6 enable topology ipv6**  
IPv6 is enabled for the IS-IS process.
- d. Run **multi-instance enable iid iid-value**  
The conventional IS-IS process is configured as an IS-IS multi-instance process.
- e. Run **quit**  
Return to the system view.
- f. Run **interface interface-type interface-number**  
The interface view is displayed.
- g. Run **ipv6 enable**  
IPv6 is enabled on the interface.
- h. Run **isis ipv6 enable [ process-id ]**  
IPv6 IS-IS is enabled on the interface, and the specified process ID is associated with the interface.
- i. Run **isis [ process-id process-id ] ipv6 link-quality low incr-cost { cost-value | max-reachable }**  
The IPv6 IS-IS interface is enabled to automatically adjust the link cost based on link quality.

#### NOTE

The *cost* parameter specifies the link cost adjustment value. After this parameter is specified:

- If the link quality changes from good to low, the link cost equals the original link cost plus the adjustment value. If the new link cost exceeds the maximum link cost allowed, the maximum link cost allowed applies:
  - The maximum link cost is 63, if the cost style is **narrow**, **narrow-compatible**, or **compatible**.
  - The maximum link cost is 16777214, if the cost style is **wide** or **wide-compatible**.
- If the link quality changes from low to good, the original link cost applies.

- j. Run **commit**

The configuration is committed.

----End

## Verifying the IPv6 IS-IS Route Selection Configuration

After configuring IPv6 IS-IS route selection, check the configurations.

## Procedure

- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv6 ] [ topology topology-name ] [ verbose | [ level-1 | level-2 ] | ipv6-**

*address [ prefix-length ] ] \* [ | count ]* command to check IS-IS routing information.

- Run the **display isis lsdb [ { level-1 | level-2 } | verbose | { local | lsp-id | is-name symbolic-name } ] \* [ process-id | vpn-instance vpn-instance-name ]** command to check information about the IS-IS LSDB.

----End

### 1.1.8.2.21 Configuring IPv6 IS-IS to Interact with Other Routing Protocols

If other routing protocols are configured on an IS-IS network, configure IS-IS to interact with these protocols for communication between them.

#### Usage Scenario

If other routing protocols are configured on an IS-IS network, note the following issues:

- Priorities of IS-IS routes

If multiple routes to the same destination are discovered by different routing protocols running on the same device, the route discovered by the protocol with the highest priority is selected. For example, if both OSPF and IS-IS are configured, the route discovered by OSPF is used because OSPF enjoys a higher priority than IS-IS by default.

If you want the route discovered by IS-IS to be preferentially selected, configure a higher priority for the route.

- Communication between areas

If other routing protocols are configured on an IS-IS network, configure IS-IS to interact with those routing protocols so that IS-IS areas can communicate with non-IS-IS areas.

##### NOTE

The LSDBs of different IS-IS processes on a device are independent of each other. Therefore, each IS-IS process on the device considers routes of the other IS-IS processes as external routes.

To ensure successful traffic forwarding, configure IS-IS to import external routes on a device (such as a Level-1-2 IS-IS router) where external routes are configured. Such configuration enables all devices in IS-IS areas to learn external routes, implementing refined control over traffic forwarding.

To ensure successful forwarding of traffic destined for IS-IS areas, enable the other routing protocols to interact with IS-IS.

#### Pre-configuration Tasks

Before configuring IPv6 IS-IS to interact with other routing protocols, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- **Configure basic IPv6 IS-IS functions.**

- Configure basic functions of other routing protocols.

## Configuring a Preference Value for IPv6 IS-IS

If there are multiple types of routes to the same destination, you can set a preference for IS-IS to enable IS-IS routes to be preferentially selected.

### Context

If multiple routes to the same destination are discovered by different routing protocols running on the same device, the route discovered by the protocol with the highest priority is selected.

For example, if both OSPFv3 and IS-IS are configured on a network, the route discovered by OSPFv3 is used because OSPFv3 has a higher priority than IS-IS by default.

You can set a preference value for IS-IS to increase the priority of IS-IS routes so that they are preferentially selected. In addition, you can configure a routing policy to increase the priority of specified IS-IS routes, without affecting route selection.

### Procedure

- Configure a preference value for IS-IS.
  - Run **system-view**  
The system view is displayed.
  - Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - Run **ipv6 preference *preference***  
The IS-IS preference value is configured.
- Configure a preference value for specified IS-IS routes.
  - Run **system-view**  
The system view is displayed.
  - Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - Run **ipv6 preference { { route-policy *route-policy-name* | route-filter *route-filter-name* } | *preference* }<sup>\*</sup>**  
A preference is configured for the matched IS-IS routes.



*preference* takes effect only on the IS-IS routes that match the specified route-policy or route-filter.

d. Run **commit**

The configuration is committed.

----End

## Configuring IPv6 IS-IS to Import External Routes

Configuring IS-IS to import external routes on a boundary device enables devices in the IS-IS domain to learn external routes and guide traffic forwarding.

### Context

After IS-IS is configured on a boundary device to advertise a default route, the traffic destined for a destination outside the domain can be diverted to the boundary device for processing. However, because other devices in the IS-IS domain do not have external routes, a large amount of traffic is forwarded to the boundary device, which overloads the boundary device.

If multiple boundary devices are deployed, optimal routes to other routing domains need to be selected. In this case, all the other devices in the IS-IS routing domain must learn all or some external routes.

Routing policies can be configured to import only the external routes that match filtering conditions or advertise only the imported routes that match filtering conditions to other IS-IS devices.

#### NOTICE

IS-IS and other dynamic routing protocols such as OSPF and BGP often import routes from each other. If no routing policy is configured or a routing policy is incorrectly configured on a device where IS-IS, OSPF, and BGP import routes from each other, a Layer 3 routing loop may occur due to a route selection result change. As a result, services are compromised. For details about the cause of the routing loop, see [Routing Loop Detection for Routes Imported to IS-IS](#).

### Procedure

- Configure IS-IS to import external routes.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - c. Configuring IS-IS to Import External IPv6 Routes
    - To import external routes to IS-IS and set a cost for them, run the **ipv6 import-route { direct | static | unr { ripng | isis | ospfv3 } [ process-id ] } | bgp [ permit-ibgp ] [ cost cost | tag tag | route-policy route-policy-name | route-filter route-filter-name | [ level-1 | level-2 | level-1-2 ] ] \* command.**

- To import external routes to IS-IS and keep the original costs, run the **ipv6 import-route { { ospfv3 | ripng | isis } [ process-id ] | bgp [ permit-ibgp ] | direct | unr } inherit-cost [ tag tag | route-policy route-policy-name | route-filter route-filter-name | [ level-1 | level-2 | level-1-2 ] ]** command. In this case, the protocol from which routes are imported cannot be **static**.

 NOTE

IS-IS advertises all imported external routes to an IS-IS routing domain by default.

If only some imported external routes need to be advertised to the IS-IS routing domain, run the **ipv6 filter-policy export** command to set a filtering policy.

d. Run **commit**

The configuration is committed.

- (Optional) Configure IS-IS to advertise some external routes to an IS-IS routing domain.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

- c. Configure IS-IS to advertise specified external routes to the IS-IS routing domain.

Run any of the following commands as required:

- Configure a basic ACL:

- 1) Run the **ipv6 filter-policy { acl6-number | acl6-name acl6-name } export [ protocol [ process-id ] ]** command.
- 2) Run **quit**  
Return to the system view.
- 3) Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**  
The ACL view is displayed.
- 4) Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment | source { source-ipv6-address { prefix-length | source-wildcard } | source-ipv6-address/prefix-length | any } | time-range time-name | [ vpn-instance vpn-instance-name | vpn-instance-any ] ]**  
A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.

- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.  
Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- Based on an IP prefix list:

Run the **ipv6 filter-policy ipv6-prefix *ipv6-prefix-name* export [ protocol [ process-id ] ]** command.

- Based on a route-policy:

Run the **ipv6 filter-policy route-policy *route-policy-name* export [ protocol [ process-id ] ]** command.

 **NOTE**

After this command is run, only external routes that meet the specified conditions can be advertised to the IS-IS routing domain.

d. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After enabling IS-IS to import routes from other protocols, check the IS-IS and IP routing tables.

## Procedure

- Run the **display isis lsdb [ { level-1 | level-2 } | verbose | { local | lsp-id | is-name *symbolic-name* } ] \* [ process-id | vpn-instance *vpn-instance-name* ]** command to check IS-IS LSDB information.

- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv6 ] [ verbose | [ level-1 | level-2 ] | ipv6-address [ prefix-length ] ] \*** command to check IS-IS routing information.
- Run the **display ipv6 routing-table ipv6-prefix ipv6-prefix-name [ verbose ]** command to check the IP routing table.

----End

### 1.1.8.2.22 Adjusting the IPv6 IS-IS Route Convergence Speed

Accelerating IS-IS route convergence can improve the fault location efficiency and network reliability.

#### Usage Scenario

The procedure for implementing IS-IS is as follows:

- Establishment of neighboring relationships: establishes neighboring relationships by exchanging Hello packets between two devices.
- LSP flooding: implements LSDB synchronization between devices in the same area.
- SPF calculation: uses the SPF algorithm to calculate IS-IS routes, and delivers the IS-IS routes to the routing table.

To accelerate the IS-IS route convergence, configure the following parameters:

- Interval for detecting IS-IS neighboring device failures.
- Flooding parameters of CSNPs and LSPs.
- Interval for SPF calculation.

You can also configure convergence priorities for IPv6 IS-IS routes so that key routes can converge first if the network topology changes, which minimizes adverse impacts on key services.

#### Pre-configuration Tasks

Before configuring the IPv6 IS-IS route convergence speed, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- [Configure basic IPv6 IS-IS functions](#).

#### Configuring the Interval for Detecting IS-IS Neighboring Device Failures

To minimize the effects caused by neighboring device failures on an IS-IS network, accelerate the speed of detecting IS-IS neighboring device failures.

#### Context

Connection status between an IS-IS device and its neighboring devices can be monitored by exchanging Hello packets at intervals. An IS-IS neighboring device is considered Down if the IS-IS device does not receive any Hello packets from the

neighboring device within a specified period (holdtime). A failure in an IS-IS neighboring device will trigger LSP flooding and SPF calculation, after which IS-IS routes re-converge.

To adjust the fault detection speed, use the following methods to accelerate the speed of detecting IS-IS neighboring device failures:

- Configure the interval at which Hello packets are sent.
- Configure the number of Hello packets that are sent before the local device considers the neighbor Down.

 NOTE

Holdtime of neighboring devices = Interval at which Hello packets are sent x Number of Hello packets that are sent before the local device considers the neighbor Down.  
The maximum value of the holdtime is 65535s.

- **Configuring Dynamic IPv6 BFD for IS-IS.**

 NOTE

Configuring IPv6 BFD for IS-IS is recommended because this method provides a faster fault detection speed than the other two methods.

## Procedure

- Set an interval at which Hello packets are sent.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface interface-type interface-number**  
The interface view is displayed.
  - c. Run **isis timer hello hello-interval [ level-1 | level-2 ] [ conservative ]**  
The interval at which Hello packets are sent is set.  
If the **conservative** parameter is specified in the command, the conservative mode is enabled for the holdtime of the IS-IS neighbor relationship.
    - If the parameter is specified and the holdtime of the IS-IS neighbor relationship is less than 20s, the IS-IS neighbor relationship is disconnected when the hold time elapses.
    - If the parameter is not specified and the holdtime of the IS-IS neighbor relationship is less than 20s, the IS-IS neighbor relationship is disconnected when the period of the hold time and a delay elapses.

 NOTE

A broadcast link can transmit both Level-1 and Level-2 Hello packets. You can set different intervals for these two types of Hello packets. By default, both Level-1 and Level-2 Hello packets are sent.

A P2P link can transmit only one type of Hello packets. Therefore, neither **level-1** or **level-2** needs to be specified if a P2P link is used.

The timer must be longer than the time a device takes to perform a master/slave main control board switchover. If the timer is set to less than the switchover time, a protocol intermittent interruption occurs during a switchover. The default timer value is recommended.

d. Run **commit**

The configuration is committed.

- Set the holding multiplier for neighboring devices.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis timer holding-multiplier number [ level-1 | level-2 ]**

The number of Hello packets that are sent before the local device considers the neighbor Down is configured.

 NOTE

A broadcast link can transmit both Level-1 and Level-2 Hello packets. You can set different intervals for these two types of Hello packets. By default, both Level-1 and Level-2 Hello packets are sent.

A P2P link can transmit only one type of Hello packets. Therefore, neither **level-1** or **level-2** needs to be specified if a P2P link is used.

d. Run **commit**

The configuration is committed.

----End

## Setting Flooding Parameters of SNPs and LSPs

To speed up LSDB synchronization between devices, set flooding parameters of SNPs and LSPs to proper values.

## Context

SNPs consist of CSNPs and PSNPs. CSNPs carry summaries of all LSPs in LSDBs, ensuring LSDB synchronization between neighboring routers. SNPs are processed differently on broadcast links and P2P links.

- On a broadcast link, CSNPs are periodically sent by a DIS device. If a router detects that its LSDB is not synchronized with that on its neighboring router, the router will send PSNPs to apply for missing LSPs.
- On a P2P link, CSNPs are sent only during initial establishment of neighbor relationships. If a request is acknowledged, a neighboring router will send a

PSNP in response to a CSNP. If a router detects that its LSDB is not synchronized with that on its neighboring router, the router will send PSNPs to apply for missing LSPs.

You can modify the following parameters of SNPs and LSPs on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X to speed up LSDB synchronization:

- **Set an interval at which CSNPs are sent.**
- **Configure the intelligent timer to control LSP generation.**
- **Set the size of an LSP.**
- **Set the LSP update interval.**
- **Set the maximum lifetime for LSPs.**
- **Set the maximum holdtime for the largest IS-IS route cost in local LSPs.**
- **Enable LSP fast flooding.**
- **Set an interval at which LSPs are retransmitted over a P2P link.**
- **Configure automatic IS-IS LSP Remaining Lifetime adjustment.**

## Procedure

- Set an interval at which CSNPs are sent.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **interface interface-type interface-number**  
The interface view is displayed.
  - c. Run **isis timer csnp csnp-interval [ level-1 | level-2 ]**  
The interval at which CSNPs are sent is set on the specified interface.
- Configure the intelligent timer to control LSP generation.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
  - c. Run **timer lsp-generation max-interval [ init-interval [ incr-interval ] ] [ level-1 | level-2 ]**  
The intelligent timer is configured to control LSP generation.  
The intelligent timer involves three parameters, and the parameters are described as follows:
    - When only *max-interval* is used, the intelligent timer becomes a one-shot timer.

- If both *init-interval* and *incr-interval* are configured, the initial interval for generating an LSP is *init-interval*, and the interval for generating an LSP with the same LSP ID for the second time is *incr-interval*. From the third time on, each time the route changes, the interval for generating an LSP doubles until the interval reaches *max-interval*. If the local routing information still changes frequently within *max-interval*, the delay remains *max-interval*. If the local routing information does not change after the interval specified by *max-interval* expires or the IS-IS process is restarted, the interval decreases to *init-interval*.
- If *init-interval* is specified but *incr-interval* is not specified, *init-interval* is used as the interval for generating an LSP for the first time, and *max-interval* is used as the interval for generating subsequent LSPs. If the local routing information changes frequently within the interval specified by *max-interval*, the delay remains *max-interval*. If the local routing information does not change after the interval specified by *max-interval* expires or the IS-IS process is restarted, the interval decreases to *init-interval*.

d. Run **commit**

The configuration is committed.

- Set the size of an LSP.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run **lsp-length originate max-size**

The size of an LSP to be generated is set.

d. Run **lsp-length receive max-size**

The size of an LSP to be received is set.

 **NOTE**

*max-size* of an LSP to be generated must be less than or equal to *max-size* of an LSP to be received.

The value of *max-size* in the **lsp-length** command must meet the following conditions.

- The MTU of an Ethernet interface must be greater than or equal to the sum of the value of *max-size* plus 3.
- The MTU of a P2P interface must be greater than or equal to the value of *max-size*.

e. Run **commit**

The configuration is committed.

- Set the LSP update interval.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run **timer lsp-refresh refresh-value**

An LSP update interval is set.

To ensure the synchronization of LSPs in the entire area, IS-IS periodically sends all the current LSPs.

 NOTE

Ensure that the LSP update interval is at least 300s shorter than the maximum LSP lifetime so that new LSPs can reach all routers in the area before the original LSPs expire.

The larger a network, the greater the deviation between the LSP update interval and the maximum LSP lifetime.

d. Run **commit**

The configuration is committed.

- Set the maximum lifetime for LSPs.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run **timer lsp-max-age max-age-value**

The maximum lifetime is set for LSPs.

When a device generates the system LSP, it fills in the maximum lifetime for this LSP. The lifetime of the LSP decreases with time. If the device does not receive any updated LSPs and the lifetime of the LSP is reduced to 0, the device keeps the LSP for another 60s. If the device fails to receive any updated LSPs within the 60s, it deletes the LSP from the LSDB.

d. Run **commit**

The configuration is committed.

- Set the maximum holdtime for the largest IS-IS route cost in local LSPs.

a. Run **system-view**

The system view is displayed.

b. Run **interface interface-type interface-number**

The interface view is displayed.

c. Run **isis [ process-id process-id ] peer hold-max-cost timer timer-val**

The holdtime of the maximum cost in local IS-IS LSPs is set.

When an IS-IS interface changes from down to up, the IS-IS neighbor relationship is re-established. After IGP route convergence is completed,

traffic is switched back. In most cases, IGP routes converge quickly. However, many services that depend on IGP routes may require a delayed switchback. To achieve this, run the **isis peer hold-max-cost** command to set the holdtime for the maximum route cost (16777214 in wide cost style and 63 in narrow cost style) in local IS-IS LSPs, so that traffic is forwarded over the original path before the hold-max-cost timer expires. After the timer expires, the normal cost value is restored, and then traffic is normally switched back.

- d. (Optional) Run **isis peer hold-cost cost-val timer timer-val**

The period during which IS-IS keeps the specified cost in local LSPs is set.

- e. Run **commit**

The configuration is committed.

- Enable LSP fast flooding.

- a. Run **system-view**

The system view is displayed.

- b. Run **interface interface-type interface-number**

The interface view is displayed.

- c. Run **isis timer lsp-throttle throttle-interval [ count count ]**

The minimum interval at which LSPs are sent by the interface and the maximum number of LSPs that can be sent within the interval are set.

The *count* parameter specifies the maximum number of LSPs that can be sent within the interval specified by *throttle-interval*. The value of *count* is an integer ranging from 1 to 1000.

- d. Run **commit**

The configuration is committed.

- Set an interval at which LSPs are retransmitted over a P2P link.

- a. Run **system-view**

The system view is displayed.

- b. Run **interface interface-type interface-number**

The interface view is displayed.

- c. (Optional) Run **isis circuit-type p2p**

The broadcast interface is simulated as a P2P interface.

An interval at which LSPs are retransmitted takes effect only on P2P interfaces. Therefore, to configure the interval on a broadcast interface, change the broadcast interface to a P2P interface first.

- d. Run **isis timer lsp-retransmit retransmit-interval**

The interval at which LSPs are retransmitted over a P2P link is set.

- e. Run **commit**

The configuration is committed.

- Configure automatic IS-IS LSP Remaining Lifetime adjustment.

- a. Run **system-view**  
The system view is displayed.
- b. Run **isis [ process-id ]**  
The IS-IS view is displayed.
- c. (Optional) Run **undo lsp-remaining-lifetime refresh disable**  
Automatic IS-IS LSP Remaining Lifetime adjustment is enabled.
- d. Run **lsp-remaining-lifetime refresh timer { refreshvalue | lsp-max-age }**  
An IS-IS LSP Remaining Lifetime value is set.
- e. Run **commit**  
The configuration is committed.

----End

## Adjusting the SPF Calculation Interval

By adjusting the SPF calculation interval, you can ensure that IS-IS responds to network changes in time and reduce the system resources consumed by SPF calculation.

### Context

When a network changes frequently, IS-IS performs SPF calculation frequently. Frequent SPF calculation will consume excessive CPU resources, affecting services.

To solve this problem, configure an intelligent timer to control the SPF calculation interval. For example, to speed up IS-IS route convergence, set the interval for SPF calculation to a small value, and set the interval to a large value after the IS-IS network becomes stable.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

#### Step 3 Run **timer spf max-interval [ init-interval [ incr-interval ] ]**

The SPF intelligent timer is configured.

The interval for SPF calculation is described as follows:

- The delay for the first SPF calculation is *init-interval*; the delay for the second SPF calculation is *incr-interval*. From the third time on, the SPF calculation delay doubles each time the route changes until the delay reaches *max-interval*. If network flapping persists within the interval specified by *max-interval*, the delay remains *max-interval*. If the network does not flap within *max-interval* or the IS-IS process is restarted, the delay decreases to *init-interval*.

- If *incr-interval* is not used, the delay for the first SPF calculation is *init-interval*. From the second time on, the delay remains *max-interval*. If the local routing information changes frequently within *max-interval*, the delay for SPF calculation remains *max-interval*. If the local routing information does not change within the interval specified by *max-interval* or the IS-IS process is restarted, the delay decreases to *init-interval*.
- When only *max-interval* is used, the intelligent timer becomes a one-shot timer.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring Convergence Priorities for IPv6 IS-IS Routes

You can set a high convergence priority for key routes on an IS-IS network to ensure that these routes converge first if the network topology changes. This minimizes the impact on important services.

### Context

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X allows you to configure the highest convergence priority for specific IS-IS routes so that those IS-IS routes converge first when a network topology changes.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run isis [ *process-id* ]

The IS-IS view is displayed.

#### Step 3 Run ipv6 prefix-priority [ **level-1** | **level-2** ] { **critical** | **high** | **medium** | **very-low** } { **ipv6-prefix** *prefix-name* | **tag** *tag-value* }

A convergence priority is set for IS-IS routes.

The rules for applying the convergence priorities of IS-IS routes are as follows:

- Existing IS-IS routes converge based on the priorities configured in the **ipv6 prefix-priority** command.
- New IS-IS routes that match the filtering rules converge based on the priorities configured in the **ipv6 prefix-priority** command.
- If an IS-IS route meets the matching rules of multiple convergence priorities, the highest convergence priority is used.
- The convergence priority of Level-1 IS-IS routes is higher than that of Level-2 IS-IS routes.
- If no level is specified, the configuration takes effect on both Level-1 and Level-2 IS-IS routes.

 NOTE

The **ipv6 prefix-priority** command is only applicable to the public network.

After the **ipv6 prefix-priority** command is run, the convergence priority of 128-bit IS-IS host routes is changed from medium to low, and the convergence priorities of the other IS-IS routes are changed based on the configuration of the **ipv6 prefix-priority** command.

**Step 4 Run commit**

The configuration is committed.

----End

## Enabling IS-IS Intelligent Convergence (IPv6)

Enabling IS-IS IPv6 intelligent convergence can speed up IS-IS route convergence, thereby improving convergence performance.

### Context

In a fault-triggered switching scenario where the local device receives a route from a single node, IS-IS intelligent convergence can be enabled to allow IS-IS to perform fast route convergence by using the fast convergence algorithm. This improves convergence performance.

### Procedure

**Step 1** Run the **system-view** command to enter the system view.

**Step 2** Run the **isis [ process-id ]** command to enter the IS-IS view.

**Step 3** Run the **ipv6 enable [ topology { compatible [ enable-mt-spf ] | ipv6 | standard } ]** command to enable IPv6 for the IS-IS process.

**Step 4** Run the **ipv6 intelligent-convergence enable** command to configure IS-IS IPv6 intelligent convergence.

 NOTE

In the same networking scenario, the convergence speeds of devices with the **ipv6 intelligent-convergence enable** command configuration increase significantly, far higher than those of the devices without this command configuration. As a result, routing loops may occur. Therefore, exercise caution when running this command.

**Step 5** Run the **commit** command to commit the configuration.

----End

## Configuring Interface Address Advertisement Suppression

Interface address advertisement suppression ensures that interface addresses can be reused.

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **interface interface-type interface-number**

The IS-IS interface view is displayed.

**Step 3** Run **isis ipv6 enable [ process-id ]**

The IPv6 of IS-IS process is enabled.

**Step 4** Run **isis ipv6 suppress-reachability**

An IS-IS interface is configured to suppress the advertisement of interface addresses.

**Step 5** Run **commit**

The configuration is committed.

----End

## Verifying the IPv6 IS-IS Route Convergence Speed Configuration

After configuring parameters to adjust the IPv6 IS-IS route convergence speed, check the configurations.

## Procedure

- Run the **display isis interface [ [ verbose | traffic-eng ] \* | tunnel ] [ process-id | vpn-instance vpn-instance-name ]** command to check IS-IS packet information.
- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] ipv6 [ topology topology-name ] [ verbose | [ level-1 | level-2 ] | ipv6-address [ prefix-length ] ] \***  command to check the priority of IS-IS routes.

----End

### 1.1.8.2.23 Enabling the Advertisement of IPv6 Delay Information

To allow IS-IS to collect and flood information about the intra-area IPv6 link delay, you can enable the advertisement of IPv6 delay information for an IS-IS process.

## Usage Scenario

Traditional path computation algorithms compute the optimal path to a destination address based on costs. However, such a computed path may not have the minimum delay. For services that have stringent requirements on delay, path computation can be performed based on the delay rather than on the cost to ensure that service traffic is transmitted along the path with the minimum delay. To meet these requirements, enable the advertisement of IPv6 delay information for an IS-IS process. After this function is enabled, IS-IS collects and floods information about the intra-area IPv6 link delay, and BGP-LS reports the information to the controller. This allows the controller to use the delay information to compute the optimal path on the P2P network.

## Pre-configuration Tasks

Before enabling the advertisement of delay information, complete the following task:

- Configure the TWAMP Light controller to detect delay information.
- Set the network type of the IS-IS interface to P2P.**

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis [ process-id ]**

An IS-IS process is created, and the view is displayed.

*process-id* specifies an IS-IS process. If the *process-id* parameter is not specified, the system creates process 1 by default. To bind an IS-IS process to a VPN instance, run the **isis process-id vpn-instance vpn-instance-name** command.

### Step 3 Run **cost-style { wide | wide-compatible | compatible }**

An IS-IS cost style is set.

### Step 4 Run **ipv6 enable [ topology { compatible [ enable-mt-spf ] | ipv6 | standard } ]**

IPv6 is enabled for the IS-IS process.

### Step 5 (Optional) Run **ipv6 metric-delay normalize interval interval-value [ offset offset-value ]**

The link delay normalization function is configured for the IS-IS process.

For the delay-based path computation algorithm, the delay differences of links are different, and the differences may be small. However, even if the delay differences are small, only one optimal path can be generated according to the existing SPF algorithm, and load balancing cannot be implemented within a certain delay tolerance range. As a result, link resources on the network cannot be fully utilized. To resolve this problem to the greatest extent, normalization processing may be performed on the link delays with a small difference or a difference within an acceptable range so that load balancing can be implemented and link resources on the network can be fully utilized. Delay variation cannot be normalized.

You can also run the **isis [ process-id process-id-value ] ipv6 metric-delay normalize interval interval-value [ offset offset-value ]** command in the IS-IS interface view to configure the link delay normalization function on an interface. If the function is configured both for an IS-IS process and an IS-IS interface, the interface's configuration takes precedence over the process's configuration.

### Step 6 Run **ipv6 traffic-eng [ level-1 | level-2 | level-1-2 ]**

IPv6 TE is enabled for a specified level of the IS-IS process.

### Step 7 Run **ipv6 metric-delay advertisement enable [ level-1 | level-2 | level-1-2 ]**

Advertisement of the maximum and minimum IPv6 delay information is configured.

### Step 8 Run **ipv6 metric-delay average advertisement enable [ level-1 | level-2 | level-1-2 ]**

The advertisement of the average IPv6 delay is enabled.

**Step 9** Run **ipv6 metric-delay variation advertisement enable [ level-1 | level-2 | level-1-2 ]**

The advertisement of IPv6 delay variation is enabled.

**Step 10** (Optional) Run **ipv6 metric-delay suppress timer *timer-value* percent-threshold *percent-value* absolute-threshold *absolute-value***

Parameters used to suppress the advertisement of IPv6 delay information are configured.

**Step 11** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display isis interface verbose traffic-eng** command to check information about IS-IS-enabled interfaces.
- Run the **display isis traffic-eng advertisements** command to check the TE information advertised by IS-IS.

### 1.1.8.2.24 Configuring an IS-IS Process to Advertise IPv6 Packet Loss Rates

Configuring an IS-IS process to advertise IPv6 packet loss rates helps ensure that service traffic is transmitted along the path with the lowest packet loss rate.

## Prerequisites

Before configuring an IS-IS process to advertise packet loss rates, complete the following tasks:

- Run the **cost-style** command to set the IS-IS cost style to wide, wide-compatible, or compatible.
- Run the **traffic-eng** command to enable IS-IS TE.

## Context

Traditional path computation algorithms compute the optimal path to a destination address based on costs. However, this path may not be the one with the lowest packet loss rate. For services that have stringent requirements on the packet loss rate, path computation can be performed based on packet loss rates instead of costs to ensure that service traffic is transmitted along the path with the lowest packet loss rate. To meet these requirements, configure an IS-IS process to advertise IPv6 packet loss rates. After this function is configured, IS-IS collects and floods information about intra-area IPv6 packet loss rates, and BGP-LS reports the information to the controller. The controller then computes the optimal path on the P2P network based on the packet loss rates.

## Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** Run the **isis [ process-id ]** command to create an IS-IS process and enter its view.
- Step 3** Run the **cost-style { wide | wide-compatible | compatible }** command to set an IS-IS cost style.
- Step 4** Run the **ipv6 traffic-eng [ level-1 | level-2 | level-1-2 ]** command to enable IPv6 TE for a specified level of the IS-IS process.
- Step 5** Run the **ipv6 metric-link-loss advertisement enable [ level-1 | level-2 | level-1-2 ]** command to configure the function of advertising IPv6 packet loss rates.
- Step 6** (Optional) Run the **ipv6 metric-link-loss suppress timer *timer-value* percent-threshold *percent-value* absolute-threshold *absolute-value*** command to configure suppression parameters for IPv6 packet loss rate advertisement.

 **NOTE**

The advertised packet loss rates may be dynamic ones measured using TWAMP Light or static ones manually configured.

The valid dynamic packet loss rates measured using TWAMP Light are advertised only to the IS-IS interface bound to the TWAMP Light test session.

After receiving packet loss rate information, IS-IS advertises and floods the information.

- Step 7** Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

After configuring the function, verify the configuration.

- Run the **display isis interface verbose traffic-eng** command to check information about IS-IS-enabled interfaces.
- Run the **display isis traffic-eng advertisements** command to check the TE information advertised by IS-IS.

### 1.1.8.2.25 Configuring IS-IS Extended Prefix Attribute Advertisement (IPv6)

After IS-IS extended prefix advertisement is configured, route origin information can be advertised to the LSDB so that the device can determine the origin of received routes.

## Usage Scenario

IS-IS defines a type of extended prefix attribute (IPv4/IPv6 Extended Reachability Attribute) sub-TLVs. These sub-TLVs are used to describe the origin of advertised routes and can be advertised in IPv4, IPv6, and locator TLVs. Based on the advertised sub-TLVs, the device can determine whether the received routes are inter-domain routes or host routes.

The extended prefix attribute flag (IPv4/IPv6 Extended Reachability Attribute Flags) is a part of the extended prefix attribute sub-TLV. You can run a command to determine whether this flag is advertised.

## Pre-configuration Tasks

Before configuring IS-IS extended prefix attribute advertisement, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure addresses for interfaces to ensure that neighboring devices are reachable at the network layer.
- **Configure basic IPv6 IS-IS functions.**

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis [ process-id ]**

An IS-IS process is created, and its view is displayed.

### Step 3 Run **ipv6 enable [ topology { compatible [ enable-mt-spf ] | ipv6 | standard } ]**

IPv6 is enabled for the IS-IS process.

### Step 4 Run **advertise prefix-attributes flags**

The IPv6 IS-IS process is enabled to advertise the extended prefix attribute flag.

### Step 5 Run **quit**

Return to the system view.

### Step 6 (Optional) Set the N flag of the extended prefix attribute on a loopback interface in the IPv6 IS-IS process to 0.

By default, if a loopback interface's route with a 128-bit subnet mask is a host route, the extended prefix N flag is set to 1. To set the extended prefix N flag to 0, perform this step.

1. Run the **interface Loopback interface-number** command to enter the loopback interface view.
2. Run the **ipv6 enable** command to enable IPv6 on the loopback interface.
3. Run the **isis ipv6 enable [ process-id ]** command to enable IPv6 IS-IS on the interface.
4. Run the **isis [ process-id process-id-value ] ipv6 prefix-attributes node-disable** command to set the N flag of the extended prefix attribute on the loopback interface in the IPv6 IS-IS process to 0.
5. Run the **quit** command to return to the system view.

### Step 7 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, run the **display isis lsdb verbose** command to view information about the extended prefix attribute flag.

### 1.1.8.2.26 Configuring IS-IS Neighbor Relationship Flapping Suppression

IS-IS neighbor relationship flapping suppression works by delaying IS-IS neighbor relationship reestablishment or setting the link cost to the maximum value.

#### Usage Scenario

If the status of an interface carrying IS-IS services alternates between up and down, the neighbor relationship frequently flaps. During the flapping, IS-IS reestablishes the neighbor relationship and recalculates routes. In this process, a large number of packets are exchanged, adversely affecting neighbor relationship stability, IS-IS services, and other IS-IS-dependent services, such as LDP and BGP. IS-IS neighbor relationship flapping suppression can address this problem by either delaying IS-IS neighbor relationship reestablishment, or setting the link cost to the maximum value to prevent service traffic from passing through flapping links.



The following steps are optional, choose them as required.

#### Pre-configuration Tasks

Before configuring IS-IS neighbor relationship flapping suppression, complete the following tasks:

- Configure an IP address for each interface to ensure that neighboring routers are reachable at the network layer.
- [Configuring Basic IPv4 IS-IS Functions](#).

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

To disable the function globally, run the **suppress-flapping peer disable** command.

##### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

To disable IS-IS neighbor relationship flapping suppression from one of the interfaces, run the **isis suppress-flapping peer disable** command.

##### Step 3 Run **isis suppress-flapping peer hold-down interval**

The Hold-down mode is configured, and its duration is set.

Flapping suppression works in either Hold-down or Hold-max-cost mode.

- Hold-down mode: In the case of frequent flooding and topology changes during neighbor relationship establishment, interfaces prevent neighbor

relationship reestablishment during Hold-down suppression, which minimizes synchronization attempts and packet exchanges.

- Hold-max-cost mode: If the traffic forwarding path changes frequently, interfaces use the maximum value (16777214 for the wide mode and 63 for the narrow mode) as the cost of the flapping link during Hold-max-cost suppression, which prevents traffic from passing through the flapping link.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

To disable this mode, run the **isis suppress-flapping peer hold-max-cost disable** command.

**Step 4** Run **isis suppress-flapping peer { detecting-interval *detecting-interval* | threshold *threshold* | resume-interval *resume-interval* } \***

Detection parameters are configured for IS-IS neighbor relationship flapping suppression.

Each IS-IS interface on which IS-IS neighbor relationship flapping suppression is enabled starts a flapping counter. If the interval between two successive neighbor status changes from Full to a non-Full state is shorter than *detecting-interval*, a valid flapping\_event is recorded, and the flapping\_count increases by 1. When the flapping\_count reaches or exceeds *threshold*, flapping suppression takes effect. If the interval between two successive neighbor status changes from Full to a non-Full state is longer than *resume-interval*, the flapping\_count is reset.

 **NOTE**

The value of *resume-interval* must be greater than that of *detecting-interval*.

**Step 5** Run **quit**

The system view is displayed.

**Step 6** Run **quit**

The user view is displayed.

**Step 7** Run **reset isis *process-id* suppress-flapping peer [ interface { *interface-name* | *interfaceType interfaceNum* } ] [ notify-peer ]**

The interface exits the neighbor flapping suppression.

Interfaces exit from flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding IS-IS process is reset.
- An IS-IS neighbor is reset using the **reset isis peer** command.
- IS-IS neighbor relationship flapping suppression is disabled globally using the **suppress-flapping peer disable** command in the IS-IS view.
- The **reset isis suppress-flapping peer** command is run to forcibly exit flapping suppression.
- The **reset isis *process-id* suppress-flapping peer [ interface *interface-type interface-number* ] notify-peer** command is run on the peer device to forcibly exit flapping suppression.

### Step 8 Run commit

The configuration is committed.

----End

## Verifying the Configuration

Run the **display isis [ process-id ] interface interfaceType interfaceNum verbose** command to check the status of IS-IS neighbor relationship flapping suppression.

### 1.1.8.2.27 Disabling IS-IS Interface Flapping Suppression

IS-IS interface flapping suppression is globally enabled by default. If this function is not needed, you can disable it.

## Usage Scenario

If IS-IS interfaces frequently alternate between up and down, the interfaces will flap, and protocol packets will be frequently exchanged, affecting IS-IS services and other services relying on IS-IS. Interface flapping suppression can address this issue. This function allows a device to delay interface state changes to up. If this function is not needed, you can disable it.

## Pre-configuration Tasks

Before configuring IS-IS interface flapping suppression, [configure basic IS-IS functions](#).

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run isis suppress-flapping interface disable

IS-IS interface flapping suppression is disabled.

### Step 3 Run commit

The configuration is committed.

----End

## Checking the Configuration

Run the **display current-configuration configuration isis** command to check the status of IS-IS interface flapping suppression.

### 1.1.8.2.28 Configuring Routing Loop Detection for Routes Imported into IS-IS

## Usage Scenario

When an IS-IS process imports routes, routing loops may occur. If a device on which routing loop detection is enabled for routes imported into IS-IS detects that

it imports a route advertised by itself, it sends this route with a large link cost to other devices. After receiving this route, these devices preferentially select other paths, preventing routing loops.

## Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** (Optional) Run the **clear route loop-detect isis alarm-state** command to exit the routing loop detection alarm state and clear related alarms.  
 **NOTE**

If the device detects an IS-IS routing loop, it reports an alarm. Because the device cannot automatically detect whether the routing loop is eliminated, you need to run this command after the routing loop is eliminated to prevent the device from advertising a large link cost for imported routes and manually clear the IS-IS routing loop alarm. If this command is executed when the routing loop has not been eliminated, the alarm is reported again.
- Step 3** Run the **route loop-detect isis enable** command to enable routing loop detection for routes imported into IS-IS.
- Step 4** Run the **commit** command to commit the configuration.

----End

### 1.1.8.2.29 Configuring IS-IS Purge Source Tracing

IS-IS purge source tracing helps improve fault locating efficiency.

#### Usage Scenario

If network-wide IS-IS LSP deletion causes network instability, source tracing must be implemented as soon as possible to locate and isolate the fault source. However, IS-IS itself does not support source tracing. A conventional solution is to isolate nodes one by one until the fault source is located, but the process is complex and time-consuming and may compromise network services. To address this problem, enable IS-IS purge source tracing.



The following steps are optional, choose them as required.

#### Pre-configuration Tasks

Before configuring IS-IS purge LSP source tracing, complete the following tasks:

- Configure IP addresses for interfaces and ensure that neighboring devices are reachable at the network layer.
- [Configure basic IS-IS functions](#).

## Procedure

- Step 1** Run **system-view**

The system view is displayed.

To disable IS-IS purge LSP source tracing globally, run the **isis purge-source-trace disable** command.

**Step 2** Run **interface interface-type interface-number**

The interface view is displayed.

To disable this function on an interface, run the **isis purge-source-trace block** command.

**Step 3** Run **quit**

Return to the system view.

**Step 4** Run **isis purge-source-trace port port-number**

A UDP port number is configured for IS-IS purge source tracing packets.

The IS-IS purge source tracing port is used to receive and send IS-IS purge source tracing packets and is identified by a UDP port number.

**Step 5** Run **quit**

Return to the user view.

**Step 6** Run **reset isis [ process-id ] purge-source-trace**

IS-IS purge source tracing is reset.

If a large number of IS-IS purge source tracing statistics exist on a device, you can run the **reset isis purge-source-trace** command to reset the statistics and restart IS-IS purge source tracing. If the command is run, all dynamic source tracing statistics are deleted, and the device needs to re-negotiate the source tracing capability with neighboring devices.

**Step 7** Run **commit**

The configuration is committed.

----End

## Checking the Configurations

Run the **display isis process-id purge-source-trace analysis-report [ level-1 | level-2 ]** command. The command output shows information about definitely and possibly faulty nodes identified by IS-IS purge source tracing.

### 1.1.8.2.30 Configuring Dynamic BFD for IPv6 IS-IS

BFD provides link failure detection featuring light load and high speed (within milliseconds). With dynamic BFD, routing protocols can dynamically trigger the establishment of BFD sessions.

## Usage Scenario

If the network requires zero packet loss and fast convergence when the link status changes, you can configure dynamic BFD on IS-IS links.

BFD needs to be configured based on the actual network environment. If the time parameters are set improperly, network flapping may occur.

## Pre-configuration Tasks

Before configuring dynamic BFD for IPv6 IS-IS, complete the following tasks:

### Configuring BFD Globally

Before configuring dynamic BFD for IS-IS, enable BFD globally.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bfd**

BFD is configured globally.

#### Step 3 Run **commit**

The configuration is committed.

----End

### Configuring BFD for an IS-IS IPv6 Process

By configuring BFD for an IPv6 IS-IS process, you can set parameters for dynamic BFD sessions and enable dynamic BFD for IPv6 IS-IS on all IPv6 IS-IS interfaces in the process.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **isis process-id**

The IS-IS view is displayed.

#### Step 3 (Optional) Run **bfd session-up check**

BFD session check is configured for the IS-IS process.

#### NOTE

When the Layer 2 network is running normally, IS-IS neighbor relationships can be established and routes can be delivered. However, if the Layer 3 network is unreachable, upper-layer traffic loss occurs. To resolve this problem, configure BFD session check for an IS-IS process using the **bfd session-up check** command. This ensures that an IS-IS neighbor relationship is established only when the BFD session is up on corresponding interfaces. After this function is configured, it applies only to the neighbor relationships to be established, not to existing neighbor relationships.

#### Step 4 Run **ipv6 bfd all-interfaces enable**

BFD is enabled for the IPv6 IS-IS process to establish BFD sessions.

BFD must be enabled globally before you perform this step. If BFD is enabled globally, this step is performed, and the IPv6 status of neighbors is up (or the DIS

is up in the case of a broadcast network), default BFD parameters are used by all interfaces in the IS-IS process to establish BFD sessions.

**Step 5** (Optional) Run **ipv6\_bfd all-interfaces { min-rx-interval receive interval | min-tx-interval transmit-interval | detect-multiplier multiplier-value | frr-binding }** \*

BFD parameters are configured to establish BFD sessions.

BFD detection intervals are calculated as follows:

- Effective local interval at which BFD packets are sent = MAX { Configured local interval at which BFD packets are sent, Configured remote interval at which BFD packets are received }
- Effective local interval at which BFD packets are received = MAX { Configured remote interval at which BFD packets are sent, Configured local interval at which BFD packets are received }
- Effective local detection interval = Effective local interval at which BFD packets are received x Configured remote detection multiplier

**Step 6** (Optional) Disable a specified interface from creating a BFD session.

After BFD is configured for an IPv6 IS-IS process, all interfaces on which neighbor relationships are up in the IPv6 IS-IS process create BFD sessions. If BFD is not required on specific interfaces, disable these interfaces from creating BFD sessions.

1. Run **quit**

Return to the system view.

2. Run **interface interface-type interface-number**

The interface view is displayed.

3. Run **isis ipv6 bfd block**

The interface is disabled from creating a BFD session.

4. Run **quit**

Return to the system view.

5. Run **isis process-id**

The IS-IS view is displayed.

**Step 7** (Optional) Run **ipv6 bfd all-interfaces incr-cost { cost-value | max-reachable } [ wtr wtr-value ]**

The IPv6 IS-IS process is enabled to adjust the cost based on the status of an associated BFD session.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

The cost restoration delay configured for BFD association on an interface takes precedence over that configured for BFD association in a process.

**Step 8** Run **commit**

The configuration is committed.

----End

## (Optional) Configuring BFD on a Specified IPv6 Interface

You can configure dynamic BFD session parameters for a specified IPv6 interface.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 (Optional) To configure BFD session check for an IS-IS process, perform the following steps:

1. Run **isis [ process-id ]**

The IS-IS view is displayed.

2. Run **bfd session-up check**

BFD session check is configured for the IS-IS process.

 NOTE

When the Layer 2 network is running normally, IS-IS neighbor relationships can be established and routes can be delivered. However, if the Layer 3 network is unreachable, upper-layer traffic loss occurs. To resolve this problem, configure BFD session check for an IS-IS process by running the **bfd session-up check** command. This ensures that an IS-IS neighbor relationship is established only when the BFD session is up on corresponding interfaces. After this function is configured, it applies only to the neighbor relationships to be established (it does not apply to existing neighbor relationships).

#### Step 3 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 4 Run **isis ipv6 bfd enable**

BFD is enabled for the IPv6 interface to establish BFD sessions.

BFD must be enabled globally before you perform this step. If BFD is enabled globally, this step is performed, and the IPv6 status of neighbors is up (or the DIS is up in the case of a broadcast network), default BFD parameters are used by this interface to establish BFD sessions.

#### Step 5 (Optional) Run **isis ipv6 bfd { min-rx-interval receive-interval | min-tx-interval transmit-interval | detect-multiplier multiplier-value | frr-binding }** \*

BFD parameters are configured to establish BFD sessions.

 NOTE

If BFD session parameters are configured for both a process and an interface, those configured for the interface take precedence and are used by this interface to establish BFD sessions.

#### Step 6 (Optional) Run **isis ipv6 bfd incr-cost { cost-value | max-reachable } [ wtr wtr-value ]**

The interface is enabled to adjust the cost based on the status of an associated BFD session.

If the function of adjusting the cost based on the status of an associated BFD session is configured both for a process and an interface, the configuration on the interface takes precedence.

The cost restoration delay configured for BFD association on an interface takes precedence over that configured for BFD association in a process.

#### Step 7 Run **commit**

The configuration is committed.

----End

### Verifying the Configuration

After configuring dynamic IPv6 BFD for IS-IS, check information about the BFD session and dynamic BFD for IS-IS on an interface.

### Prerequisites

Dynamic IPv6 BFD for IS-IS has been configured.

### Procedure

- Run the **display isis ipv6 bfd [ process-id | vpn-instance vpn-instance-name ] session { all | peer ipv6-address | interface interface-type interface-number }** command to check information about BFD sessions.
- Run the **display isis ipv6 bfd [ process-id | vpn-instance vpn-instance-name ] interface** command to check BFD configurations on an interface.

----End

#### 1.1.8.2.31 Configuring Track BFD for IPv6 IS-IS

Track BFD allows you to manually bind an IS-IS interface to a link-bundle BFD session by specifying a session name. This function can quickly detect link faults and prevent BFD session faults caused by faults of the board on which a trunk member interface resides.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bfd**

BFD is enabled.

#### Step 3 Run **quit**

Return to the system view.

#### Step 4 Run **isis [ process-id ]**

The IS-IS view is displayed.

#### Step 5 Run **network-entity net-addr**

A network entity title (NET) is configured.

**Step 6** Run **quit**

Return to the system view.

**Step 7** Run **interface interface-type interface-number**

An interface is created.

**Step 8** Run **quit**

Return to the system view.

**Step 9** Run **bfd sess-name bind link-bundle [ compatible-mode ] peer-ipv6 ipv6-address [ vpn-instance vpn-name ] interface interface-type interface-number source-ipv6 ipv6-address**

A BFD for link-bundle session is created to detect Eth-Trunk faults, and the BFD session view is displayed.

**Step 10** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 11** Run **ipv6 enable**

IPv6 is enabled.

**Step 12** Run **isis ipv6 enable [ process-id ]**

IS-IS is enabled on the interface.

**Step 13** Run **isis ipv6 bfd track session-name bfd-session-name**

Track BFD is enabled on the interface.

**Step 14** Run **commit**

The configuration is committed.

----End

### 1.1.8.2.32 Configuring IS-IS IPv6 MT to Isolate Multicast Services from Unicast Services

On an IS-IS IPv6 network, IS-IS MT improves usage of network resources by allowing services to run in multiple logical topologies.

#### Usage Scenario

On a traditional IPv6 network, only one unicast forwarding table is available in the forwarding plane because only one unicast topology exists. Therefore, traffic of all services shares the same hop-by-hop forwarding behavior as long as the destination IP addresses of the traffic are the same. This means that various end-to-end services (for example, voice services and data services) share the same physical links. As a result, some links may be heavily congested whereas some other links are relatively idle. In addition, QoS requirements vary according to services. The traditional unicast topology cannot meet various QoS requirements.

IS-IS MT allows multiple separate logical topologies to be deployed in an IS-IS autonomous system (AS). Separate multicast topologies can be set up for multicast services, and this configuration isolates multicast topologies from unicast topologies.

IS-IS MT allows users to configure the network flexibly, reducing network construction cost.

To configure IS-IS MT, perform the following steps. First, associate an IS-IS process with topology instances so that the SPF calculation can be performed for the topology instances in the IS-IS process. Second, associate a specified interface with the topology instances so that every link can participate in the SPF calculation.

## Pre-configuration Tasks

Before configuring IS-IS MT to isolate IPv6 multicast services from IPv6 unicast services, complete the following tasks:

### Enabling MT for an IPv6 IS-IS Process

Based on service requirements and the network plan, an IS-IS process can be associated with various topology instances so that multiple logical topologies are deployed in an IS-IS AS.

### Context

You can associate an IS-IS process with unicast or multicast topology instances as required so that the SPF calculation can be performed for the topology instances in the IS-IS process. Then, you can configure parameters, such as the interface cost and protocol priority for the topology instances.

The configuration in an IS-IS topology instance view takes effect only on the topology instance. Therefore, you can configure different features and parameters for different topology instances as required.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ipv6 topology *topology-name***

An IPv6 topology instance is configured.

#### Step 3 Run **isis [ *process-id* ]**

The IS-IS view is displayed.

#### Step 4 Run **cost-style { wide | wide-compatible }**

The cost style of packets sent and received by the router is set to **wide** or **wide-compatible**.

 NOTE

Go to Step 5 or Step 6 as required.

- If the router supports only unicast services, go to Step 5.
- If the router supports only multicast services, go to Step 6.
- If the router supports both unicast and multicast services, perform the following steps.

**Step 5** To associate an IS-IS process with a topology instance, perform the following operations as required:

- To associate the IS-IS process with a specified unicast topology instance and enter the IS-IS IPv6 unicast topology view, run the **ipv6 topology topology-name topology-id** command.
- To associate the IS-IS process with a specified multicast topology instance and enter the IS-IS multicast topology view, run the **ipv6 topology topology-name topology-id multicast** command.

**Step 6** (Optional) Configure IS-IS features for IPv6 topology instances using the following commands:

- To set a preference for IS-IS, run the **ipv6 preference** command.
- To enable an IS-IS device to generate a default route, run the **default-route-advertise** command.
- To enable a device to import routes from another routing protocol into IS-IS, run the **ipv6 import-route** command in the IS-IS view.
- To control IS-IS route leaking from a Level-1 area to a Level-2 area, run the **ipv6 import-route isis level-1 into level-2** command.
- To control IS-IS route leaking from a Level-2 area to a Level-1 area, run the **ipv6 import-route isis level-2 into level-1** command.
- To enable IS-IS summarization, run the **summary** command.

**Step 7** Run **commit**

The configuration is committed.

----End

## Enabling MT for an IPv6 IS-IS Interface

After IS-IS MT is enabled, associate a specified interface with MT instances.

## Context

Various topology instances can be associated with one interface so that specified links of the interface can participate in the SPF calculation for the topology instances.

The following parameters can be configured for an interface in different topology instances:

- Cost of the interface in each topology instance
- Administrative tag of the interface in each topology instance

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

### Step 3 Run **ipv6 topology topology-name enable**

An IS-IS IPv6 topology instance is enabled for the interface.

### Step 4 Run **isis ipv6 topology topology-name**

The interface is associated with the IS-IS IPv6 topology instance.

#### NOTE

If you want to associate an interface with multiple IPv6 topology instances, repeat this step and specify different topology names.

### Step 5 (Optional) Run **isis ipv6 topology topology-name cost cost-value [ level-1 | level-2 ]**

A cost is configured for the interface in the IS-IS topology instance.

A cost change on an interface will trigger route reselection. Therefore, change the cost during network planning.

### Step 6 (Optional) Run **isis ipv6 topology topology-name tag-value tag [ level-1 | level-2 ]**

An administrative tag is configured for the interface in the IS-IS topology instance.

An administrative tag is carried in routes and used by route-policies to filter routes.

### Step 7 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of IS-IS IPv6 Multi-Topologies Used to Isolate Multicast Services from Unicast Services

After configuring IPv6 IS-IS MT to isolate multicast services from unicast services, check the IS-IS MT configurations.

## Prerequisites

IPv6 IS-IS MT has been configured to isolate multicast services from unicast services.

## Procedure

- Run the **display isis peer [ verbose ] [ process-id | vpn-instance vpn-instance-name ]** command to check information about IS-IS neighbors.

- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] ipv6 [ topology topology-name ] [ verbose | [ level-1 | level-2 ] | ip-address [ mask | mask-length ] ]** \* command to check IS-IS routing information.
- Run the **display isis spf-tree [ systemid systemid ] [ [ level-1 | level-2 ] | ipv6 | topology topology-name ]** \* **verbose [ process-id | vpn-instance vpn-instance-name ]** command to check IS-IS SPT information.

----End

### 1.1.8.2.33 Configuring IS-IS MT to Isolate IPv4 Services from IPv6 Services

On an IS-IS network, IS-IS MT improves usage of network resources by allowing multiple logical topologies to be deployed.

#### Usage Scenario

IPv4 and IPv6 share the same topology structure. An IPv4/IPv6 topology is an integrated topology in which IPv4 and IPv6 use the same shortest path in the SPF calculation. Therefore, the topology information of IPv6 must be the same as that of IPv4 when an IPv4/IPv6 topology is deployed.

However, in real-world situations, IPv4 and IPv6 may be deployed differently, and the topology information of IPv6 may differ from that of IPv4. In the scenario where an IPv4/IPv6 topology is deployed, some routers and links do not support IPv6, and they cannot receive IPv6 packets from the routers supporting the IPv4/IPv6 dual stack. As a result, these IPv6 packets are discarded. Similarly, IPv4 packets cannot be forwarded if routers and links that do not support IPv4 on the network. When various end-to-end services, such as voice and data services, share the same physical links, either IPv4 or IPv6 packets are discarded on the shared links, affecting transmission quality.

IS-IS MT allows multiple separate logical topologies to be deployed in an IS-IS AS, and the SPF calculation can be performed in an IPv6 topology independently. In addition, IPv6 forwarding tables can be created for the IPv6 topology.

IS-IS MT allows users to configure the network flexibly, reducing network construction cost.

To configure IS-IS MT, perform the following steps. First, associate an IS-IS process with topology instances so that the SPF calculation can be performed for the topology instances in the IS-IS process. Second, associate a specified interface with the topology instances so that every link can participate in the SPF calculation.

#### Enabling MT for an IS-IS Process

Based on service requirements and the network plan, an IS-IS process can be associated with various topology instances so that multiple logical topologies are deployed in an IS-IS AS.

#### Context

You can associate an IS-IS process with IPv4 or IPv6 topology instances as required so that the SPF calculation can be performed for the topology instances in the IS-IS process. Then, configure parameters, you can such as the interface cost and protocol priority for the topology instances.

The configuration in an IS-IS topology instance view takes effect only on the topology instance. Therefore, you can configure different features and parameters for different topology instances as required.

## Procedure

- Enable MT for an IS-IS IPv4 process.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **ip topology topology-name**

An IPv4 topology instance is created.
  - c. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - d. Run **cost-style { wide | wide-compatible }**

The cost style of the packets to be sent and accepted by the router is set to **wide** or **wide-compatible**.
  - e. Run **topology topology-name [ topology-id { multicast | topology-id } ]**

The IS-IS process is associated with a specified IPv4 topology instance, and the IPv4 topology view is displayed.
  - f. (Optional) Configure IS-IS parameters in the topology instance. The following commands are supported:
    - To set a preference for IS-IS, run the **preference** command.
    - To enable an IS-IS device to generate a default route, run the **default-route-advertise** command.
    - To enable the device to import routes from another routing protocol, run the **import-route** command in the IS-IS view.
    - To control IS-IS route leaking from a Level-1 area to a Level-2 area, run the **import-route isis level-1 into level-2** command.
    - To control IS-IS route leaking from a Level-2 area to a Level-1 area, run the **import-route isis level-2 into level-1** command.
  - g. Run **commit**

The configuration is committed.
- Enable MT for an IS-IS IPv6 process.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **ipv6 topology topology-name**

An IPv6 topology instance is created.
  - c. Run **isis [ process-id ]**

The IS-IS view is displayed.
  - d. Run **cost-style { wide | wide-compatible }**

The cost style of the packets to be sent and accepted by the router is set to **wide** or **wide-compatible**.

- e. Run **ipv6 topology topology-name [ topology-id { multicast | topology-id } ]**

The IS-IS process is associated with a specified IPv6 topology instance, and the IPv6 topology view is displayed.

- f. (Optional) Configure IS-IS parameters in the topology instance. The following commands are supported:

- To set a preference for IS-IS, run the **preference** command.
- To enable an IS-IS device to generate a default route, run the **default-route-advertise** command.
- To enable the device to import routes from another routing protocol, run the **import-route** command in the IS-IS view.
- To control IS-IS route leaking from a Level-1 area to a Level-2 area, run the **import-route isis level-1 into level-2** command.
- To control IS-IS route leaking from a Level-2 area to a Level-1 area, run the **import-route isis level-2 into level-1** command.

- g. Run **commit**

The configuration is committed.

----End

## Enabling MT for an IS-IS Interface

After IS-IS MT is enabled, associate a specified interface with MT instances.

### Context

Various topology instances can be associated with one interface so that specified links of the interface can participate in the SPF calculation for the topology instances.

The following parameters can be configured for an interface in different topology instances:

- Cost of the interface in each topology instance
- Administrative tag of the interface in each topology instance

An interface transmitting different services needs to be associated with different topology instances. Perform the following operations:

- [Associate an interface with an IPv4 topology instance](#).
- [Associate an interface with an IPv6 topology instance](#).

### Procedure

#### Step 1 Associate an interface with an IPv4 topology instance.

1. Run **system-view**

The system view is displayed.

2. Run **interface interface-type interface-number**  
The interface view is displayed.
3. Run **ip topology topology-name enable**  
An IPv4 topology instance is bound to the interface.
4. Run **isis topology topology-name**  
An IPv4 topology instance is enabled on the interface.
5. (Optional) Run **isis topology topology-name cost cost [ level-1 | level-2 ]**  
A cost is configured on the interface for the IS-IS IPv4 topology instance.  
A cost change on an interface will trigger route reselection. Therefore, change the cost during network planning.
6. (Optional) Run **isis topology topology-name tag-value tag [ level-1 | level-2 ]**  
An administrative tag is configured on the interface for the IS-IS IPv4 topology instance.  
An administrative tag is carried in routes and used by route-policies to filter routes.
7. Run **commit**  
The configuration is committed.

**Step 2** Associate an interface with an IPv6 topology instance.

1. Run **system-view**  
The system view is displayed.
2. Run **interface interface-type interface-number**  
The interface view is displayed.
3. Run **ipv6 topology topology-name enable**  
An IPv6 topology instance is bound to the interface.
4. Run **isis ipv6 topology topology-name**  
An IPv6 topology instance is enabled on the interface.
5. (Optional) Run **isis ipv6 [ topology topology-name ] cost cost-value [ level-1 | level-2 ]**  
A cost is configured on the interface for the IS-IS IPv6 topology instance.  
A cost change on an interface will trigger route reselection. Therefore, change the cost during network planning.
6. (Optional) Run **isis ipv6 [ topology topology-name ] tag-value tag [ level-1 | level-2 ]**  
An administrative tag is configured on the interface for the IS-IS IPv6 topology instance.  
An administrative tag is carried in routes and used by route-policies to filter routes.

## 7. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration of IS-IS Multi-Topologies Used to Isolate IPv4 Services from IPv6 Services

After configuring IPv4 IS-IS MT to isolate IPv4 services from IPv6 services, check the configuration about IS-IS multi-topologies.

## Prerequisites

IPv4 IS-IS MT has been configured to isolate IPv4 services from IPv6 services.

## Procedure

- Run the **display isis peer [ verbose ] [ process-id | vpn-instance vpn-instance-name ]** command to check information about IS-IS neighbors.
- Run the **display isis route [ process-id | vpn-instance vpn-instance-name ] [ ipv4 | ipv6 ] [ topology topology-name ] [ verbose | [ level-1 | level-2 ] ]** command to check IS-IS routing information.
- Run the **display isis spf-tree [ systemid systemid ] [ [ level-1 | level-2 ] | ipv6 | topology topology-name ] \* verbose [ process-id | vpn-instance vpn-instance-name ]** command to check IS-IS SPT information.

----End

### 1.1.8.2.34 Configuring IPv6 IS-IS Route Summarization

On a medium-or large-scale IS-IS network, a large number of IS-IS IPv6 routing entries may exist. Configuring route summarization helps reduce the IPv6 routing table size, which in turn speeds up route lookup.

## Usage Scenario

The IS-IS routing table of a device on a medium or large IS-IS network contains a large number of routing entries. Storing the routing table consumes a large number of memory resources, and transmitting and processing routing information consume lots of network resources. IS-IS route summarization can solve this problem.

Routes with the same IPv6 prefix can be summarized into one route. This greatly reduces the number of routing entries and minimizes system resource consumption. In addition, if a specific link frequently alternates between up and down, the link state changes will not be advertised to devices that are located beyond the summary route network segment, preventing route flapping and improving network stability.

## Pre-configuration Tasks

Before configuring IPv6 IS-IS route summarization, complete the following tasks:

- Configure IPv6 addresses for interfaces to ensure that neighboring devices are reachable at the network layer.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

### Step 3 Run **ipv6 summary ipv6-address prefix-length [ explicit ] [ avoid-feedback | generate\_null0\_route | tag tag-value | learning-avoid-loop | [ level-1 | level-1-2 | level-2 ] ] \***

IS-IS is configured to generate an IPv6 summary route.

#### NOTE

- After IPv6 route summarization is configured, the IPv6 routing table on the local IS-IS device still contains the specific routes that participate in the summarization.
- The IPv6 routing tables on other IS-IS devices that receive the LSPs sent by the local device contain the summary route rather than the specific routes.
- If both **generate\_null0\_route** and **learning-avoid-loop** are specified and the device learns a route with the same prefix from another device and generates a loop-free path, the device preferentially selects the learned route. Otherwise, a blackhole route is generated.

### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display isis route** command to check summary routes in the IS-IS routing table.
- Run the **display ipv6 routing-table [ verbose ]** command to check summary routes in the IPv6 routing table.

### 1.1.8.2.35 Configuring IPv6 IS-IS Auto FRR

IPv6 IS-IS Auto FRR can rapidly switch traffic from a faulty link to a backup link, ensuring that the traffic interruption time is less than 50 ms and improving IS-IS network reliability.

## Usage Scenario

As networks develop, Voice over Internet Protocol (VoIP) and online video services pose increasingly higher requirements for real-time transmission. Nevertheless, if an IS-IS fault occurs, multiple operations, including fault detection, LSP updating, LSP flooding, route calculation, and forward information base (FIB) entry delivery,

must be performed to switch traffic to a new link. These operations take a long time, much longer than 50 ms, the minimum delay to which users are sensitive. As a result, user experience is affected.

With IPv6 IS-IS Auto FRR, devices can rapidly switch traffic from a faulty link to an alternate one without interrupting the traffic, which significantly improves IS-IS network reliability.

IPv6 IS-IS Auto FRR is applicable to the services that are sensitive to packet delay and packet loss.

 NOTE

IPv6 IS-IS Auto FRR can only take effect in the standard topology.

## Pre-configuration Tasks

Before configuring IPv6 IS-IS Auto FRR, complete the following tasks:

- Configure the link layer protocol on interfaces.
- Configure IP addresses for interfaces and ensure that neighboring devices are reachable at the network layer.
- **Configure basic IPv6 IS-IS functions.**
- Configure a local LDP session on the source node, the PQ node, and the nodes between them if remote LFA FRR is required.

## Procedure

- Configure the LFA algorithm.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **isis [ process-id ]**  
The IS-IS process is enabled, and the IS-IS view is displayed.
  - c. Run **ipv6 frr**  
The IS-IS IPv6 FRR view is displayed.
  - d. (Optional) Run **ecmp disable { level-1 | level-2 }**  
IPv6 IS-IS ECMP FRR is disabled.

With ECMP FRR, the IGP pre-calculates backup paths for load balancing links based on the LSDBs on the entire network and stores the backup paths in the forwarding table. The backup paths are used for traffic protection in the case of link failures. If the network topology changes, ECMP FRR triggers calculation of new backup paths, which increases the system calculation pressure. In addition, the backup entries generated by the ECMP FRR function increase the memory usage of the system. In this case, you can run this command to disable the ECMP FRR function.

- e. Run **loop-free-alternate [ level-1 | level-2 | level-1-2 ]**

IPv6 IS-IS Auto FRR is enabled to generate loop-free backup links.

If no level is specified, IPv6 IS-IS Auto FRR is enabled for both Level-1 and Level-2.

- f. (Optional) Run **frr-policy route route-policy-name**  
A route-policy is configured to filter backup routes so that only the matched backup routes are added to the routing table.
  - g. (Optional) Run **tiebreaker { node-protecting | lowest-cost | non-ecmp | srlg-disjoint | hold-max-cost } preference preference [ level-1 | level-2 | level-1-2 ]**  
The solution of selecting a backup path for IPv6 IS-IS Auto FRR is set.
  - h. Run **commit**  
The configuration is committed.
- Configure the remote LFA algorithm.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **isis [ process-id ]**  
The IS-IS process is enabled, and the IS-IS view is displayed.
    - c. Run **frr**  
The IS-IS FRR view is displayed.
    - d. Run **remote-lfa tunnel ldp [ maximum-reachable-cost cost-value ] [ level-1 | level-2 | level-1-2 ]**  
Remote LFA Auto FRR is enabled.
    - e. Run **quit**  
Return to the IS-IS view.
    - f. Run **ipv6 frr**  
The IS-IS IPv6 FRR view is displayed.
    - g. (Optional) Run **ecmp disable { level-1 | level-2 }**  
IPv6 IS-IS ECMP FRR is disabled.  
By default, IPv6 IS-IS ECMP FRR is enabled.  
With ECMP FRR, an IGP pre-calculates backup paths for load balancing links based on the LSDBs on the entire network and saves the backup paths in the forwarding table for traffic protection in the case of a fault. If the network topology changes, ECMP FRR triggers calculation of new backup paths, which increases the system calculation pressure. In addition, the backup entries generated by the ECMP FRR function increase the memory usage of the system. In this case, you can run this command to disable ECMP FRR.
    - h. (Optional) Run **remote-lfa tunnel ldp over-ipv4 [ level-1 | level-2 | level-1-2 ]**  
Remote LFA Auto FRR is enabled.  
LFA FRR cannot be used to calculate backup links on large-scale networks, especially on ring networks. As a result, reliability requirements cannot be met. In this case, IPv6 IS-IS Remote LFA FRR can be implemented. To enable IPv6 IS-IS Remote LFA, run the **remote-lfa**

command. Remote LFA computes a PQ node based on the protection path and establishes a tunnel between the source and PQ node to provide backup next-hop protection. If the protection link fails, traffic is automatically switched to the tunnel backup path, improving network reliability.

- i. (Optional) Run **tiebreaker { node-protecting | lowest-cost | non-ecmp | srlg-disjoint } preference preference [ level-1 | level-2 | level-1-2 ]**

A solution for selecting a backup path for IPv6 IS-IS auto FRR is set.

By default, the solution of selecting a backup path for IPv6 IS-IS auto FRR is node-protection path first.

- j. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring IPv6 IS-IS Auto FRR, check the configurations.

- Run the **display isis route [ level-1 | level-2 ] [ process-id | vpn-instance vpn-instance-name ] ipv6 [ ipv6-address [ prefix-length ] ] [ verbose ]** command to check information about the primary and backup links after IPv6 IS-IS Auto FRR is enabled.
- Run the **display isis [ process-id ] srlg { srlgGroupId | all }** command to check shared risk link group information.

### 1.1.8.2.36 Configuring an IS-IS Link Group

If one of the links that are load-balancing traffic fails, traffic may get lost. To prevent traffic loss, configure an IS-IS link group.

## Usage Scenario

A link group is a group of links that are used for traffic forwarding. If one (or more) of the links fails and the bandwidth of the other links in the group is not sufficient to carry the traffic, the link group automatically increases the link cost so that the routes along its member links are not selected. Then, traffic is switched to a backup link, which prevents traffic loss. If the link failure is cleared, the link group can automatically restore the original link cost. After the link cost is restored, routes are recalculated, and traffic is then forwarded along the optimal route.

## Pre-configuration Tasks

Before configuring an IS-IS link group, configure [basic IPv4 IS-IS functions](#) or [basic IPv6 IS-IS functions](#).

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

The IS-IS view is displayed.

**Step 3** Run **link-group group-name**

A link group is created, and the IS-IS link group view is displayed.

**Step 4** (Optional) Run **cost-offset { cost | max-reachable | maximum }**

The device is configured to automatically adjust the link costs of member links in the link group.



If **maximum** is specified in the command and automatic cost adjustment is triggered, the links in the link group are used to transmit TE information, but not used for route calculation.

**Step 5** (Optional) Run **min-members min-num**

The device is configured to automatically adjust the costs of all links in the link group when the number of available member links in the link group falls below *min-num*.

**Step 6** (Optional) Run **revert-members revert-num**

The device is configured to automatically restore the original costs of all links in the link group when the number of member links in the link group is greater than or equal to *revert-num*.



The value of *revert-num* must be greater than or equal to that of *min-number*.

**Step 7** Run **quit**

Exit the IS-IS link group view.

**Step 8** Run **quit**

Exit the IS-IS view.

**Step 9** Run **interface interface-type interface-number**

The interface view is displayed.

**Step 10** Run **isis [ ipv6 [ topology topology-name ] ] link-group group-name [ level-1 | level-2 ]**

The interface is bound to the link group.



To bind multiple interfaces to the link group, repeat steps **9** and **10**.

**Step 11** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

An IS-IS link group has been configured.

- Run the **display isis link-group [process-id] [group-name]** command to check IS-IS link-group information.
- Run the **display isis link-group interface interface-type interface-number** command to check information about the link group to which the IS-IS interface has been bound.

### 1.1.8.2.37 Configuring IS-IS Local MT

With IS-IS local multicast topology (MT), multicast traffic can be transmitted over a traffic engineering (TE) tunnel, and the router through which the tunnel passes can generate multicast forwarding entries.

#### Usage Scenario

When both multicast and an MPLS TE tunnel are deployed on a network, multicast packets may be forwarded through the TE tunnel. As a result, the routers spanned by the TE tunnel cannot detect the multicast packets and therefore cannot create multicast forwarding entries. To prevent this problem, configure local multicast topology (MT) and enable IGP Shortcut for the TE tunnel. In this case, an MIGP routing table can be established to guide multicast packet forwarding.



Local MT takes effect only in IS-IS processes of the public network instance.

#### Pre-configuration Tasks

Before configuring IS-IS local MT, complete the following tasks:

- Configure IP addresses for interfaces to ensure network connectivity between neighboring nodes.

#### Enabling Local MT

After local MT is enabled, routers through which a TE tunnel passes can generate multicast forwarding entries.

#### Context

Enable local MT in the IS-IS system view before configuring IS-IS MT. To enable the router through which an IGP Shortcut-enabled TE tunnel passes to generate multicast forwarding entries, perform the following operations on the router.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **isis [ process-id ]**

The IS-IS view is displayed.

**Step 3** Run **cost-style { narrow | wide | wide-compatible | { compatible | narrow-compatible } [ relax-spf-limit ] }**

The cost style is configured.

 **NOTE**

MPLS TE features take effect only when the cost style is **compatible**, **wide**, or **wide-compatible**.

**Step 4** Run **traffic-eng [ level-1 | level-2 | level-1-2 ]**

TE is configured for the process.

**Step 5** Run **local-mt enable**

Local MT is enabled.

**Step 6** Run **commit**

The configuration is committed.

----End

## (Optional) Controlling the Size of the MIGP Routing Table

You can configure a filtering policy based on the multicast source address so that the router adds only the routes destined to the specified multicast source address to the independent Multicast IGP (MIGP) routing table, which controls the size of the MIGP routing table.

## Context

After local MT is enabled in an IS-IS system, IS-IS calculates routes and creates an MIGP routing table. When the next-hop outbound interface calculated by IS-IS is an IGP Shortcut-enabled TE tunnel interface, the router uses a physical interface as the next-hop outbound interface and saves it in the MIGP routing table. To control the number of entries in the MIGP routing table, speed up the MIGP routing table lookup, and reduce memory resource consumption, you can configure a multicast source address-based filtering policy so that the device adds only the matched routes to the MIGP routing table.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

The IS-IS view is displayed.

**Step 3** Configure a local MT routing policy.

Run any of the following commands as required:

- Based on the basic ACL:

- a. Run **local-mt filter-policy acl { acl-number | acl-name }**  
A policy is configured for IS-IS local MT.
- b. Run **quit**  
Return to the system view.
- c. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**  
The ACL view is displayed.
- d. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***  
A rule is configured for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- Based on the IP prefix:

Run **local-mt filter-policy ip-prefix ip-prefix-name**

A routing policy allows only the routes with the destination address being the multicast source address to be added to the MIGP routing table, reducing CPU and memory usage.

#### Step 4 Run commit

The configuration is committed.

----End

### Verifying the Configuration of IS-IS Local MT

After configuring local MT, check the MIGP routing table, routing information, SPF tree, and IS-IS statistics.

### Prerequisites

Local MT has been configured.

### Procedure

- Run the **display isis [ process-id ] migp-routing [ ip-address [ mask | mask-length ] | [ level-1 | level-2 ] | verbose ]** command to check the IS-IS MIGP routing table.

----End

### 1.1.8.2.38 Improving IS-IS Network Security

On a network that requires high security, you can configure IS-IS authentication.

### Usage Scenario

IS-IS authentication or optional checksum can improve IS-IS network security.

- IS-IS authentication encapsulates authentication information into Hello packets, Link State Protocol Data Units (LSPs), and Sequence Number Protocol Data Units (SNPs). After an IS-IS device receives the packets, it checks whether the encapsulated authentication information is correct. The IS-IS device only accepts the packets with correct authentication information. The authentication mechanism enhances IS-IS network security. IS-IS authentication consists of area authentication, routing domain authentication, and interface authentication.  
IS-IS authentication ensures that the data is correctly transmitted at the network layer.
- IS-IS optional checksum encapsulates checksum Type-Length-Values (TLVs) into SNPs and Hello packets. After an IS-IS device receives the packets, it checks whether the checksum TLVs are correct. The IS-IS device only accepts the packets with correct checksum TLVs. The authentication mechanism enhances IS-IS network security.  
IS-IS optional checksum ensures that the data is correctly transmitted at the link layer.

### Pre-configuration Tasks

Before configuring IS-IS authentication, complete the following tasks:

- Configure IP addresses for interfaces to ensure that neighboring nodes are reachable at the network layer.
- Configure basic IPv4 IS-IS functions.**
- Configure basic IPv6 IS-IS functions.**
- Configure a keychain

## Configuring IS-IS Authentication

After IS-IS authentication is configured, authentication information can be encapsulated into LSPs, SNPs, or Hello packets to ensure packet transmission security.

### Context

Generally, IS-IS packets do not carry authentication information, and received packets are not authenticated. To prevent malicious packets from attacking the network, configure IS-IS authentication to improve network security.

- Area authentication: Authentication passwords are encapsulated into IS-IS packets in Level-1 areas. The receiver only accepts the packets that have been authenticated. Therefore, you need to configure IS-IS area authentication to authenticate packets in Level-1 areas.
- Routing domain authentication: Authentication passwords are encapsulated into IS-IS packets in Level-2 areas. The receiver only accepts the packets that have been authenticated. Therefore, you need to configure IS-IS routing domain authentication to authenticate packets in Level-2 areas.
- Interface authentication: The authentication information is encapsulated into IS-IS Hello packets. A neighbor relationship can be established only after IS-IS Hello packets are authenticated. Therefore, you need to configure interface authentication to authenticate neighbors.

#### NOTE

When configuring IS-IS authentication, the authentication mode and passwords of the routers in the same area must be consistent so that IS-IS packets can be flooded normally.

An IS-IS neighbor relationship cannot be established if interface authentication fails. An IS-IS neighbor relationship can be established regardless of whether IS-IS area or routing domain authentication succeeds.

When configuring an authentication password, select the ciphertext mode because the password is saved in the configuration file in simple text if you select simple text mode, which has a high risk. To ensure device security, change the password periodically.

For security purposes, you are advised not to use weak security algorithms in IS-IS. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

### Procedure

- Configure IS-IS area authentication.
  - Run **system-view**  
The system view is displayed.
  - Run **isis [ process-id ]**  
The IS-IS view is displayed.

- c. Run one of the following commands based on the encryption type:
  - **area-authentication-mode { simple { plain plain | [ cipher ] cipher } | md5 { [ cipher ] cipher | plain plain } } [ ip | osi ] [ snp-packet { authentication-avoid | send-only } | all-send-only ]**
  - **area-authentication-mode keychain keychain-name [ snp-packet { authentication-avoid | send-only } | all-send-only ]**
  - **area-authentication-mode hmac-sha256 key-id key-id{ plain plain | [ cipher ] cipher } [ snp-packet { authentication-avoid | send-only } | all-send-only ]**

The area authentication mode is configured for IS-IS.

#### NOTICE

After the **area-authentication-mode** command is run, IS-IS discards the locally stored Level-1 LSPs that fail authentication and newly received Level-1 LSPs and SNPs that fail authentication after they are automatically aged. To prevent packet loss before authentication is configured, you can specify the **send-only** parameter in the command when deploying authentication on a network running services.

The MD5 algorithm is not recommended if high security is required. To prevent routing information from being tampered with, you are advised to enable authentication and use high-security algorithms such as HMAC-SHA256 to improve security.

You can configure area authentication in any of the following ways:

- Do not specify **snp-packet** or **all-send-only**. In this case, the device encapsulates authentication information in the LSPs and SNPs to be sent, authenticates received LSPs and SNPs, and discards the LSPs and SNPs that fail to be authenticated.
- Specify **snp-packet authentication-avoid**. In this case, the device encapsulates authentication information in the LSPs to be sent and authenticates received LSPs; the device neither encapsulates authentication information in the SNPs to be sent nor authenticates received SNPs.
- Specify **snp-packet send-only**. In this case, the device encapsulates authentication information both in the LSPs and SNPs to be sent, but authenticates only received LSPs (not received SNPs).
- Specify **all-send-only**. In this case, the device encapsulates authentication information both in the LSPs and SNPs to be sent, but does not authenticate received LSPs or SNPs.

- d. Run **commit**

The configuration is committed.

- Configure IS-IS routing domain authentication.

a. Run **system-view**

The system view is displayed.

b. Run **isis [ process-id ]**

The IS-IS view is displayed.

c. Run one of the following commands based on the encryption type:

- **domain-authentication-mode { simple { plain *plain* | cipher *cipher* } | md5 { [ cipher ] *cipher* | plain *plain* } } [ ip | osi ] [ snp-packet { authentication-avoid | send-only } | all-send-only ]**
- **domain-authentication-mode keychain *keychain-name* [ snp-packet { authentication-avoid | send-only } | all-send-only ]**
- **domain-authentication-mode hmac-sha256 key-id *key-id* { plain *plain* | [ cipher ] *cipher* } [ snp-packet { authentication-avoid | send-only } | all-send-only ]**

The routing domain authentication mode is configured for IS-IS.

**NOTICE**

After the **domain-authentication-mode** command is run, IS-IS discards the locally stored Level-2 LSPs that fail authentication and newly received Level-2 LSPs and SNPs that fail authentication after they are automatically aged. To prevent packet loss before authentication is configured, you can specify the **send-only** parameter in the command when deploying authentication on a network running services.

The MD5 algorithm is not recommended if high security is required. To prevent routing information from being tampered with, you are advised to enable authentication and use high-security algorithms such as HMAC-SHA256 to improve security.

You can configure routing domain authentication in any of the following ways:

- Do not specify **snp-packet** or **all-send-only**. In this case, the device encapsulates authentication information in the LSPs and SNPs to be sent, authenticates received LSPs and SNPs, and discards the LSPs and SNPs that fail to be authenticated.
- Specify **snp-packet authentication-avoid**. In this case, the device encapsulates authentication information in the LSPs to be sent and authenticates received LSPs; the device neither encapsulates authentication information in the SNPs to be sent nor authenticates received SNPs.
- Specify **snp-packet send-only**. In this case, the device encapsulates authentication information both in the LSPs and SNPs to be sent, but authenticates only received LSPs (not received SNPs).

- Specify **all-send-only**. In this case, the device encapsulates authentication information both in the LSPs and SNPs to be sent, but does not authenticate received LSPs or SNPs.
- d. Run **commit**

The configuration is committed.
- Configure IS-IS interface authentication.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **interface interface-type interface-number**

The interface view is displayed.
  - c. Run one of the following commands based on the encryption type:
    - **isis authentication-mode { simple { plain plain | cipher cipher } | md5 { [ cipher ] cipher | plain plain } } [ level-1 | level-2 ] [ ip | osi ] [ send-only ]**
    - **isis authentication-mode keychain keychain-name [ level-1 | level-2 ] [ send-only ]**
    - **isis authentication-mode hmac-sha256 key-id key-id{ plain plain | [ cipher ] cipher } [ level-1 | level-2 ] [ send-only ]**

The IS-IS authentication mode and password are configured on the interface.

#### NOTE

The MD5 algorithm is not recommended if high security is required. To prevent routing information from being tampered with, you are advised to enable authentication and use high-security algorithms such as HMAC-SHA256 to improve security.

When you select parameters, note the following rules:

- If **send-only** is specified, the device encapsulates authentication information to Hello packets to be sent but does not authenticate received Hello packets. The neighbor relationships can be set up when the authentication is not required or packets are authenticated.
- If **send-only** is not configured, ensure that passwords of all interfaces with the same level in the same network are consistent.
- **Level-1 areas** and **level-2** can be set only on Ethernet interfaces.
- When IS-IS interfaces are Level-1-2 interfaces and **Level-1 areas** or **level-2** is not specified in the command, authentication modes and passwords are configured for both Level-1 areas and Level-2 Hello packets.

- d. Run **commit**

The configuration is committed.

----End

## Configuring the Optional Checksum

The optional checksum encapsulates optional checksum Type-Length-Values (TLVs) into SNPs and Hello packets. After an IS-IS device receives the packets, it checks whether the checksum TLVs are correct, which improves network security.

### Context

The optional checksum encapsulates optional checksum TLVs into the Complete Sequence Numbers Protocol Data Units (CSNPs), Partial Sequence Number Protocol Data Units (PSNPs), and Hello packets sent by IS-IS devices. When the peer device receives the encapsulated packets, it checks whether TLVs carried in the packets are correct. If TLVs are not correct, the peer device discards the packets for network security.

### Procedure

**Step 1** Run system-view

The system view is displayed.

**Step 2** Run isis

An IS-IS process is created, and the IS-IS view is displayed.

**Step 3** Run **optional-checksum enable**

IS-IS optional checksum is enabled.



If MD5 authentication or Keychain authentication with valid MD5 authentication is configured on an IS-IS interface or area, IS-IS devices send Hello packets and SNP packets without checksum TLVs but verify the checksum of received packets.

**Step 4** Run commit

The configuration is committed.

----End

## Verifying the Configuration of Improving IS-IS Network Security

After improving IS-IS network security, check the information about IS-IS neighbors to determine whether the IS-IS authentication succeeds.

### Prerequisites

Configurations have been performed to improve IS-IS network security.

### Procedure

**Step 1** Run the **display isis lsdb verbose** command to check the information about IS-IS LSDB.

----End

### 1.1.8.2.39 Configuring Whitelist Session-CAR for IS-IS

You can configure whitelist session-CAR for IS-IS to isolate bandwidth resources by session for IS-IS packets. This configuration prevents bandwidth preemption among IS-IS sessions in the case of a traffic burst.

#### Context

When IS-IS packets suffer a traffic burst, bandwidth may be preempted among different IS-IS sessions. To resolve this problem, you can configure whitelist session-CAR for IS-IS to isolate bandwidth resources by session. If the default parameters of whitelist session-CAR for IS-IS do not meet service requirements, you can adjust them as required.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **whitelist session-car isis { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } \***

Parameters of whitelist session-CAR for IS-IS are configured.

In normal cases, you are advised to use the default values of these parameters.

##### Step 3 (Optional) Run **whitelist session-car isis disable**

Whitelist session-CAR for IS-IS is disabled.

By default, whitelist session-CAR for IS-IS is enabled. In normal cases, you are advised to keep this function enabled. Disable it if it becomes abnormal or adversely affects other services.

##### Step 4 Run **commit**

The configuration is committed.

----End

#### Verifying the Configuration

After configuring whitelist session-CAR for IS-IS, verify the configuration.

Run the **display cpu-defend whitelist-l2 session-car isis statistics slot slot-id** command to check statistics about whitelist session-CAR for IS-IS on a specified interface board.

If you want to check such statistics within a specific period, first you can run the **reset cpu-defend whitelist-l2 session-car isis statistics slot slot-id** command to clear statistics about whitelist session-CAR for IS-IS on the specified interface board. After some time, run the **display cpu-defend whitelist-l2 session-car isis statistics slot slot-id** command again.

### 1.1.8.2.40 Configuring Micro-Isolation CAR for IS-IS

#### Context

By default, micro-isolation CAR is enabled for IS-IS, implementing micro-isolation protection for IS-IS packets based on interfaces and destination MAC addresses to protect session establishment. When IS-IS packets encounter a traffic burst, a large number of IS-IS packets may preempt interface bandwidth resources. In the case of an attack, a large number of invalid packets sent to other MAC addresses may preempt the bandwidth. Therefore, you are advised not to disable this function.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **micro-isolation protocol-car isis { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } \***

Micro-isolation CAR parameters are configured for IS-IS.

In normal cases, you are advised to use the default values of these parameters. *pir-value* must be greater than or equal to *cir-value*, and *pbs-value* must be greater than or equal to *cbs-value*.

##### Step 3 (Optional) Run **micro-isolation protocol-car isis disable**

Micro-isolation CAR is disabled for IS-IS.

By default, micro-isolation CAR is enabled for IS-IS. To disable this function, run the **micro-isolation protocol-car isis disable** command. If this function is disabled, micro-isolation protection is no longer implemented for IS-IS packets based on interfaces and destination MAC addresses. In normal cases, you are advised to keep micro-isolation CAR enabled for IS-IS.

##### Step 4 Run **commit**

The configuration is committed.

----End

### 1.1.8.2.41 Maintaining IS-IS

Maintaining IS-IS involves resetting IS-IS, clearing IS-IS statistics, and debugging IS-IS.

#### Resetting IS-IS

Resetting an IS-IS process clears all the data of the IS-IS process and re-establishes the adjacency.

## Context

### NOTICE

Resetting an IS-IS process may cause a service interruption. Therefore, exercise caution when running this command.

## Procedure

- Run the **reset isis all [ process-id | vpn-instance vpn-instance-name ]** or **reset isis process-id all** command to restart the IS-IS process.
- Run the **reset isis peer system-id [ process-id | vpn-instance vpn-instance-name ]** or **reset isis process-id peer system-id** command to reset the IS-IS neighbor relationship.

----End

## Suppressing IS-IS

You can disable an IS-IS process temporarily by suppressing IS-IS without affecting IS-IS configurations.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **isis [ process-id ]**

An IS-IS process is started, and the IS-IS view displayed.

### Step 3 Run **shutdown**

The IS-IS process is disabled temporarily.

After the IS-IS process is disabled temporarily, you can still perform IS-IS configurations but the configurations do not take effect. You can run the **undo shutdown** command to cancel the suppression.

### Step 4 Run **commit**

The configuration is committed.

----End

## Disabling IS-IS Memory Overload Control

IS-IS memory overload control is enabled by default and can be disabled.

## Context

If the system memory is overloaded, each module needs to take necessary measures to control the memory usage increase or even reduce the memory usage. In this case, the IS-IS module takes any of the following measures to

improve IS-IS resilience: restrict the establishment of new neighbor relationships, set the Overload bit in LSP fragment zero to be sent and reject new LSPs from neighbors, restrict the installation of newly imported routes, or delete imported routes.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 (Optional) Run **isis memory-overload exception-threshold discard new-lsp**

IS-IS is enabled to set the Overload bit in its LSP fragment zero (to be sent) to 1 and discard new LSPs (received) if the system memory is overloaded and the memory usage of the IS-IS LSDB component is not among the top 3.

If the system memory is overloaded and the memory usage of the IS-IS LSDB component is among the top 3, IS-IS sets the Overload bit in its LSP fragment zero (to be sent) to 1 and discards new LSPs (received) by default to improve IS-IS resilience. However, if the memory usage of the IS-IS LSDB component is not among the top 3 and new IS-IS services are added, the IS-IS LSDB component will consume more memory resources, aggravating the system memory overload issue. To prevent this issue, run the **isis memory-overload exception-threshold discard new-lsp** command, which enables IS-IS to set the Overload bit in its LSP fragment zero (to be sent) to 1 and discard new LSPs (received). To disable this function, run the **undo isis memory-overload exception-threshold discard new-lsp** command.

IS-IS memory overload control is enabled by default. If you disable this function by performing the next step, the configuration in this step will become invalid.

### Step 3 Run **isis memory-overload control disable**

IS-IS memory overload control is disabled.

To minimize the impact of memory overload on services, you are advised to keep IS-IS memory overload control enabled.

### Step 4 Run **commit**

The configuration is committed.

----End

## Disabling IS-IS CPU Overload Control

IS-IS CPU overload control is enabled by default and can be disabled.

## Context

By default, IS-IS CPU overload control is enabled. If a device's CPU is overloaded, each module takes necessary measures to control its own CPU usage accordingly. Upon receiving a CPU overload notification from the system, the IS-IS module controls the speeds of some internal computing processes and the establishment of neighbor relationships based on the CPU overload condition to enhance the resilience of IS-IS. In this case, new neighbor relationships cannot be established, and the established neighbor relationships are not affected.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run isis cpu-overload control disable

IS-IS CPU overload control is disabled.



To minimize the impact of CPU overload upon services, you are advised to keep IS-IS CPU overload control enabled.

### Step 3 Run commit

The configuration is committed.

----End

## (Optional) Configuring IS-IS to Discard a Specified LSP

If an LSP needs to be discarded, you can configure the IS-IS process to discard it.

## Context

IS-IS can be configured to discard a specified LSP in the following scenarios:

1. When devices on the entire network restart repeatedly due to abnormal LSPs and you have located the LSP that causes protocol restarts, you can configure this function as a last resort to prevent the device from restarting continuously. However, if this function is incorrectly configured, routing loops may occur.
2. If an LSP is identified as an attack packet, which is not supposed to appear in the local area, and the LSP has caused serious problems, such as device restarts, you can configure this function to filter out the LSP under the condition that the attack source cannot be located temporarily and that the LSP does not affect topology path computation.
3. If an LSP is identified as an attack packet, which is not supposed to appear in the local area, and it affects topology path computation and has caused serious problems, such as network-wide device restarts, you can configure this function on each device to discard the LSP to prevent it from participating in network-wide calculation.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run isis [ process-id ]

The IS-IS view is displayed.

### Step 3 Run ignore-receive-lsp { system-id sysid | lsp-id lpid }

The device is configured to discard a specified LSP.

 NOTE

If this command is incorrectly configured, services cannot be restored even if the **undo ignore-receive-lsp { system-id sysid | lsp-id lpid }** command is run. In this case, you may need to reset the process to restore services.

To filter out the LSP that affects topology path computation, you must ensure that it is removed from all the LSDBs on the entire network. Otherwise, routing loops may occur.

You are advised not to run this command to filter out the LSPs that exist on the network as running this command may filter out normal LSPs.

**Step 4 Run commit**

The configuration is committed.

----End

#### 1.1.8.2.42 IS-IS Configuration Examples

This section describes IS-IS configuration examples.

#### Example for Configuring Basic IS-IS Functions

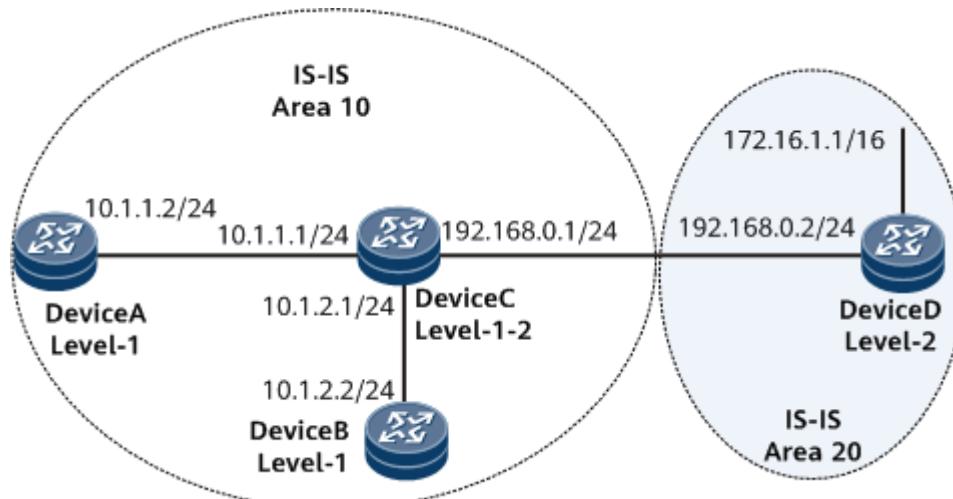
This section describes how to configure basic IS-IS functions, including specifying the NET, configuring the IS-IS level, and enabling IS-IS on each device.

#### Networking Requirements

In [Figure 1-279](#):

- Device A, Device B, Device C, and Device D run IS-IS for IP interworking.
- Device A, Device B, and Device C belong to Area 10, and Device D belongs to Area 20.
- Device A and Device B are Level-1 devices; Device C is a Level-1-2 device; Device D is a Level-2 device.

**Figure 1-279** Configuring basic IS-IS functions



| Device Name | Interface | IP Address     |
|-------------|-----------|----------------|
| Device A    | GE1/0/0   | 10.1.1.2/24    |
| Device B    | GE1/0/0   | 10.1.2.2/24    |
| Device C    | GE1/0/0   | 10.1.1.1/24    |
|             | GE2/0/0   | 10.1.2.1/24    |
|             | GE3/0/0   | 192.168.0.1/24 |
| Device D    | GE1/0/0   | 192.168.0.2/24 |
|             | GE2/0/0   | 172.16.1.1/16  |

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable IS-IS, configure the level, and specify the NET on each router.
2. Configure Device A and Device C to authenticate Hello packets in a specified mode and with the specified password.
3. View the IS-IS LSDB and the routing table of each router.

## Data Preparation

To complete the configuration, you need the following data:

- Area addresses of Device A, Device B, Device C, and Device D
- Levels of Device A, Device B, Device C, and Device D

## Procedure

### Step 1 Configure an IP address for each interface.

Configure an IP address for each [Figure 1-279](#) interface. For detailed configurations, see Configuration Files.

### Step 2 Configure basic IS-IS functions.

# Configure Device A.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-1
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] is-level level-1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
```

```
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] isis circuit-type p2p
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

# Configure Device C.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] is-level level-1-2
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] isis circuit-type p2p
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] isis enable 1
[*DeviceC-GigabitEthernet2/0/0] isis circuit-type p2p
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] interface gigabitethernet 3/0/0
[*DeviceC-GigabitEthernet3/0/0] isis enable 1
[*DeviceC-GigabitEthernet3/0/0] isis circuit-type p2p
[*DeviceC-GigabitEthernet3/0/0] commit
[~DeviceC-GigabitEthernet3/0/0] quit
```

# Configure Device D.

```
[~DeviceD] isis 1
[*DeviceD-isis-1] is-level level-2
[*DeviceD-isis-1] network-entity 20.0000.0000.0004.00
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis enable 1
[*DeviceD-GigabitEthernet2/0/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] isis enable 1
[*DeviceD-GigabitEthernet1/0/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet1/0/0] commit
[~DeviceD-GigabitEthernet1/0/0] quit
```

**Step 3** Configure the authentication mode and password used by Device A and Device C to authenticate Hello packets.

# Configure Device A.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] isis authentication-mode hmac-sha256 key-id 1 cipher YsHsjx_202206
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure Device C.

```
[~DeviceC] interface gigabitethernet 1/0/0
[~DeviceC-GigabitEthernet1/0/0] isis authentication-mode hmac-sha256 key-id 1 cipher YsHsjx_202206
[*DeviceC-GigabitEthernet1/0/0] commit
[~DeviceC-GigabitEthernet1/0/0] quit
```

**Step 4** Verify the configuration.

# Display the IS-IS LSDB of each router.

```
[~DeviceA] display isis lsdb
Database information for ISIS(1)

Level-1 Link State Database
LSPID Seq Num Checksum Holdtime Length ATT/P/OL
```

```

0000.0000.0001.00-00* 0x00000006 0xbff7d 649 68 0/0/0
0000.0000.0002.00-00 0x00000003 0xef4d 545 68 0/0/0
0000.0000.0003.00-00 0x00000008 0x3340 582 111 1/0/0
Total LSP(s): 3
*(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
ATT-Attached, P-Partition, OL-Overload
[~DeviceB] display isis lsdb
Database information for ISIS(1)

Level-1 Link State Database
LSPID Seq Num Checksum Holdtime Length ATT/P/OL

0000.0000.0001.00-00 0x00000006 0xbff7d 642 68 0/0/0
0000.0000.0002.00-00* 0x00000003 0xef4d 538 68 0/0/0
0000.0000.0003.00-00 0x00000008 0x3340 574 111 1/0/0
Total LSP(s): 3
*(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
ATT-Attached, P-Partition, OL-Overload
[~DeviceC] display isis lsdb
Database information for ISIS(1)

Level-1 Link State Database
LSPID Seq Num Checksum Holdtime Length ATT/P/OL

0000.0000.0001.00-00 0x00000006 0xbff7d 638 68 0/0/0
0000.0000.0002.00-00 0x00000003 0xef4d 533 68 0/0/0
0000.0000.0003.00-00* 0x00000008 0x3340 569 111 1/0/0
Total LSP(s): 3
*(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
ATT-Attached, P-Partition, OL-Overload
Level-2 Link State Database
LSPID Seq Num Checksum Holdtime Length ATT/P/OL

0000.0000.0003.00-00* 0x00000008 0x55bb 650 100 0/0/0
0000.0000.0004.00-00 0x00000005 0x651 629 84 0/0/0
Total LSP(s): 2
*(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
ATT-Attached, P-Partition, OL-Overload
[~DeviceD] display isis lsdb
Database information for ISIS(1)

Level-2 Link State Database
LSPID Seq Num Checksum Holdtime Length ATT/P/OL

0000.0000.0003.00-00 0x00000008 0x55bb 644 100 0/0/0
0000.0000.0004.00-00* 0x00000005 0x651 624 84 0/0/0
Total LSP(s): 2
*(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
ATT-Attached, P-Partition, OL-Overload
```

# Display the IS-IS routing table of each router. In the routing table of a Level-1 device, there must be a default route with a Level-1-2 device as the next hop. A Level-2 device must have all Level-1 and Level-2 routes.

```
[~DeviceA] display isis route
Route information for ISIS(1)

ISIS(1) Level-1 Forwarding Table

IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

10.1.1.0/24 10 NULL GE1/0/0 Direct D/-L/-/
10.1.2.0/24 20 NULL GE1/0/0 10.1.1.1 A/-/-/-
192.168.0.0/24 20 NULL GE1/0/0 10.1.1.1 A/-/-/-
0.0.0.0/0 10 NULL GE1/0/0 10.1.1.1 A/-/-/-
Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
U-Up/Down Bit Set
Protect Type: L-Link Protect, N-Node Protect
```

```
[~DeviceC] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 10.1.1.0/24 10 NULL GE1/0/0 Direct D/-L/-/
 10.1.2.0/24 10 NULL GE2/0/0 Direct D/-L/-/
 192.168.0.0/24 10 NULL GE3/0/0 Direct D/-L/-/
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
 ISIS(1) Level-2 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 10.1.1.0/24 10 NULL - Direct D/-L/-/
 10.1.2.0/24 10 NULL - Direct D/-L/-/
 192.168.0.0/24 10 NULL - Direct D/-L/-/
 172.16.0.0/16 20 NULL GE3/0/0 192.168.0.2 A/-/-/-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
[~DeviceD] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-2 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 192.168.0.0/24 10 NULL GE3/0/0 Direct D/-L/-/
 10.1.1.0/24 20 NULL GE3/0/0 192.168.0.1 A/-/-/-
 10.1.2.0/24 20 NULL GE3/0/0 192.168.0.1 A/-/-/-
 172.16.0.0/16 10 NULL GE2/0/0 Direct D/-L/-/
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
is-level level-1
network-entity 10.0000.0000.0001.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1
isis circuit-type p2p
isis authentication-mode hmac-sha256 key-id 1 cipher %^%#c;\wJ4Qi8l1FMGM}KmlK9rha/.D!
$"~0(Ep66z~%^%#

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
is-level level-1
network-entity 10.0000.0000.0002.00
```

```

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
isis enable 1
isis circuit-type p2p

return
```

- Device C configuration file

```

sysname DeviceC

isis 1
is-level level-1-2
network-entity 10.0000.0000.0003.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1
isis circuit-type p2p
isis authentication-mode hmac-sha256 key-id 1 cipher %^%#c;\wJ4Qi8l1FMGM}KmlK9rha/.D!
$%^~0(Ep66z~%^%#

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
isis enable 1
isis circuit-type p2p

interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
isis enable 1
isis circuit-type p2p

return
```

- Device D configuration file

```

sysname DeviceD

isis 1
is-level level-2
network-entity 20.0000.0000.0004.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.0.2 255.255.255.0
isis enable 1
isis circuit-type p2p

interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.1 255.255.0.0
isis enable 1
isis circuit-type p2p

return
```

## Example for Configuring IS-IS DIS Election

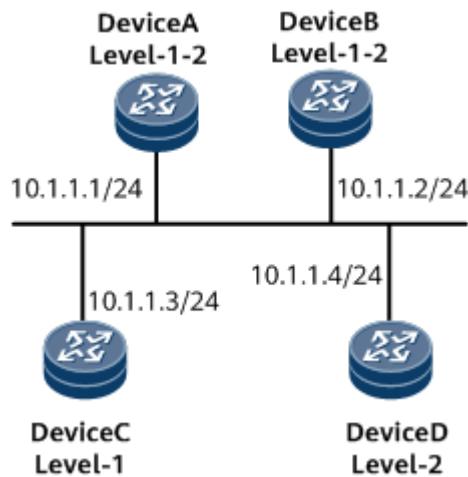
This section describes how to configure the IS-IS DIS election, including configuring basic IS-IS functions and configuring the DIS priority on each device.

## Networking Requirements

In [Figure 1-280](#):

- Device A, Device B, Device C, and Device D run IS-IS for IP interworking.
- Device A, Device B, Device C, and Device D belong to Area 10, and the network type is broadcast (Ethernet in this case).
- Device A and Device B are Level-1-2 devices; Device C is a Level-1 device; Device D is a Level-2 device.
- The DIS priority of the interface on Device A is 100.
- It is required to change the DIS priority so that DeviceA functions as a Level-1-2 DIS.

**Figure 1-280** Configuring IS-IS DIS election



| Device Name | Interface | IP Address  |
|-------------|-----------|-------------|
| Device A    | GE1/0/0   | 10.1.1.1/24 |
| Device B    | GE1/0/0   | 10.1.1.2/24 |
| Device C    | GE1/0/0   | 10.1.1.3/24 |
| Device D    | GE1/0/0   | 10.1.1.4/24 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable IS-IS and specify the NET on each router for interworking.

2. View information about the IS-IS interface on each router with the default priority.
3. Configure the DIS priority of the interface on each router.

## Data Preparation

To complete the configuration, you need the following data:

- Area addresses of the four routers
- Levels of the four routers
- DIS priority of the interface on Device A

## Procedure

**Step 1** Configure an IPv4 address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Display the MAC address of the GE interface on each router. When the DIS priority of each interface is the same, the router whose interface has the largest MAC address is elected as the DIS.

# Display the MAC address of GigabitEthernet 1/0/0 on Device A.

```
[~DeviceA] display arp interface gigabitethernet 1/0/0
ARP timeout:1200s
IP ADDRESS MAC ADDRESS EXPIRE(M) TYPE INTERFACE VPN-INSTANCE
VLAN/CEVLAN PVC

10.1.1.1 00e0-fc10-afec I GE1/0/0

Total:1 Dynamic:0 Static:0 Interface:1 Remote:0
Redirect:0
```

# Display the MAC address of GigabitEthernet 1/0/0 on Device B.

```
[~DeviceB] display arp interface gigabitethernet 1/0/0
ARP timeout:1200s
IP ADDRESS MAC ADDRESS EXPIRE(M) TYPE INTERFACE VPN-INSTANCE
VLAN/CEVLAN PVC

10.1.1.2 00e0-fccd-acdf I GE1/0/0

Total:1 Dynamic:0 Static:0 Interface:1 Remote:0
Redirect:0
```

# Display the MAC address of GigabitEthernet 1/0/0 on Device C.

```
[~DeviceC] display arp interface gigabitethernet 1/0/0
ARP timeout:1200s
IP ADDRESS MAC ADDRESS EXPIRE(M) TYPE INTERFACE VPN-INSTANCE
VLAN/CEVLAN PVC

10.1.1.3 00e0-fc50-25fe I GE1/0/0

Total:1 Dynamic:0 Static:0 Interface:1 Remote:0
Redirect:0
```

# Display the MAC address of GigabitEthernet 1/0/0 on Device D.

```
[~DeviceD] display arp interface gigabitethernet 1/0/0
ARP timeout:1200s
IP ADDRESS MAC ADDRESS EXPIRE(M) TYPE INTERFACE VPN-INSTANCE
VLAN/CEVLAN PVC
```

```

10.1.1.4 00e0-fcf0-23ed I GE1/0/0

Total:1 Dynamic:0 Static:0 Interface:1 Remote:0
Redirect:0
```

**Step 3** Enable IS-IS.

# Configure Device A.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

# Configure Device C.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] is-level level-1
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] commit
[~DeviceC-GigabitEthernet1/0/0] quit
```

# Configure Device D.

```
[~DeviceD] isis 1
[*DeviceD-isis-1] network-entity 10.0000.0000.0004.00
[*DeviceD-isis-1] is-level level-2
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] isis enable 1
[*DeviceD-GigabitEthernet1/0/0] commit
[~DeviceD-GigabitEthernet1/0/0] quit
```

# Display information about IS-IS neighbors of Device A.

```
[~DeviceA] display isis peer
Peer information for ISIS(1)

System Id Interface Circuit Id State HoldTime Type PRI

0000.0000.0002 GE1/0/0 0000.0000.0002.01 Up 30s L1(L1L2) 64
0000.0000.0003 GE1/0/0 0000.0000.0002.01 Up 30s L1 64
0000.0000.0002 GE1/0/0 0000.0000.0004.01 Up 30s L2(L1L2) 64
0000.0000.0004 GE1/0/0 0000.0000.0004.01 Up 30s L2 64

Total Peer(s): 4
```

# Display information about the IS-IS interface on Device A.

```
[~DeviceA] display isis interface
Interface information for ISIS(1)
```

| Interface | Id  | IPV4.State | IPV6.State          | MTU  | Type  | DIS   |
|-----------|-----|------------|---------------------|------|-------|-------|
| GE1/0/0   | 001 | Up         | Mtu:Up/Lnk:Dn/IP:Dn | 1497 | L1/L2 | No/No |

# Display information about the IS-IS interface on Device B.

|                                   |                                               |
|-----------------------------------|-----------------------------------------------|
| [~DeviceB]                        | <b>display isis interface</b>                 |
| Interface information for ISIS(1) |                                               |
| Interface                         | Id IPV4.State IPV6.State MTU Type DIS         |
| GE1/0/0                           | 001 Up Mtu:Up/Lnk:Dn/IP:Dn 1497 L1/L2 Yes/Yes |

# Display information about the IS-IS interface on Device D.

|                                   |                                              |
|-----------------------------------|----------------------------------------------|
| [~DeviceD]                        | <b>display isis interface</b>                |
| Interface information for ISIS(1) |                                              |
| Interface                         | Id IPV4.State IPV6.State MTU Type DIS        |
| GE1/0/0                           | 001 Up Mtu:Up/Lnk:Dn/IP:Dn 1497 L1/L2 No/Yes |

As shown in the preceding interface information, when the default DIS priority is used, the interface on Device B has the largest MAC address among the interfaces on Level-1 devices. Therefore, Device B is elected as the Level-1 DIS. The interface on Device D has the largest MAC address among the interfaces on Level-2 devices. Therefore, Device D is elected as the Level-2 DIS. Level-1 and Level-2 pseudo nodes are 0000.0000.0002.01 and 0000.0000.0004.01, respectively.

#### Step 4 Configure the DIS priority on Device A.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] isis dis-priority 100
[*DeviceA-GigabitEthernet1/0/0] commit
```

# Display information about IS-IS neighbors of Device A.

|                              |                                              |
|------------------------------|----------------------------------------------|
| [~DeviceA]                   | <b>display isis peer</b>                     |
| Peer information for ISIS(1) |                                              |
| System Id                    | Interface Circuit Id State HoldTime Type PRI |
| 0000.0000.0002               | GE1/0/0 0000.0000.0001.01 Up 27s L1(L1L2) 64 |
| 0000.0000.0003               | GE1/0/0 0000.0000.0001.01 Up 27s L1 64       |
| 0000.0000.0002               | GE1/0/0 0000.0000.0001.01 Up 27s L2(L1L2) 64 |
| 0000.0000.0004               | GE1/0/0 0000.0000.0001.01 Up 27s L2 64       |
| Total Peer(s): 4             |                                              |

#### Step 5 Verify the configuration.

# Display information about the IS-IS interface on Device A.

|                                   |                                               |
|-----------------------------------|-----------------------------------------------|
| [~DeviceA]                        | <b>display isis interface</b>                 |
| Interface information for ISIS(1) |                                               |
| Interface                         | Id IPV4.State IPV6.State MTU Type DIS         |
| GE1/0/0                           | 001 Up Mtu:Up/Lnk:Dn/IP:Dn 1497 L1/L2 Yes/Yes |

As shown in the preceding information, after the DIS priority of the IS-IS interface is changed, DeviceA immediately becomes a Level-1-2 DIS and the pseudonode is 0000.0000.0001.01.

# Display information about IS-IS neighbors and the IS-IS interface on Device B.

|                              |                                               |
|------------------------------|-----------------------------------------------|
| [~DeviceB]                   | <b>display isis peer</b>                      |
| Peer information for ISIS(1) |                                               |
| System Id                    | Interface Circuit Id State HoldTime Type PRI  |
| 0000.0000.0001               | GE1/0/0 0000.0000.0001.01 Up 26s L1(L1L2) 100 |

```
0000.0000.0003 GE1/0/0 0000.0000.0001.01 Up 26s L1 64
0000.0000.0001 GE1/0/0 0000.0000.0001.01 Up 26s L2(L1L2) 100
0000.0000.0004 GE1/0/0 0000.0000.0001.01 Up 26s L2 64
```

Total Peer(s): 4

[~DeviceB] **display isis interface**

Interface information for ISIS(1)

| Interface | Id  | IPV4.State | IPV6.State          | MTU  | Type  | DIS   |
|-----------|-----|------------|---------------------|------|-------|-------|
| GE1/0/0   | 001 | Up         | Mtu:Up/Lnk:Dn/IP:Dn | 1497 | L1/L2 | No/No |

# Display information about IS-IS neighbors and the IS-IS interface on Device D.

[~DeviceD] **display isis peer**

Peer information for ISIS(1)

| System Id      | Interface | Circuit Id        | State | HoldTime | Type | PRI |
|----------------|-----------|-------------------|-------|----------|------|-----|
| 0000.0000.0001 | GE1/0/0   | 0000.0000.0001.01 | Up    | 20s      | L2   | 100 |
| 0000.0000.0002 | GE1/0/0   | 0000.0000.0001.01 | Up    | 20s      | L2   | 64  |

Total Peer(s): 2

[~DeviceD] **display isis interface**

Interface information for ISIS(1)

| Interface | Id  | IPV4.State | IPV6.State          | MTU  | Type  | DIS   |
|-----------|-----|------------|---------------------|------|-------|-------|
| GE1/0/0   | 001 | Up         | Mtu:Up/Lnk:Dn/IP:Dn | 1497 | L1/L2 | No/No |

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
network-entity 10.0000.0000.0001.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1
isis dis-priority 100 level-1
isis dis-priority 100 level-2

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
network-entity 10.0000.0000.0002.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1

return
```

- Device C configuration file

```

sysname DeviceC

isis 1
is-level level-1
```

```
network-entity 10.0000.0000.0003.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.3 255.255.255.0
isis enable 1
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
isis 1
is-level level-2
network-entity 10.0000.0000.0004.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.4 255.255.255.0
isis enable 1
#
return
```

## Example for Configuring IS-IS Load Balancing

This section describes how to configure IS-IS load balancing.

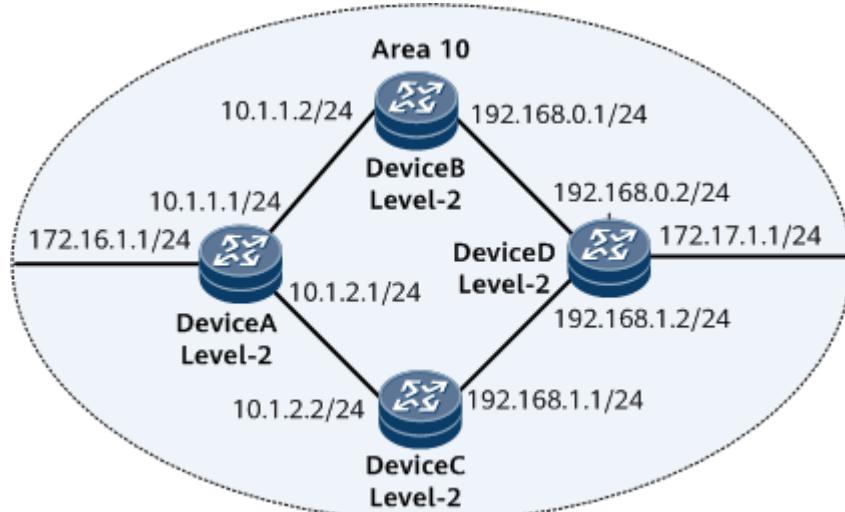
### Networking Requirements

In [Figure 1-281](#):

- Device A, Device B, Device C, and Device D run IS-IS for IP interworking.
- Device A, Device B, Device C, and Device D are Level-2 devices in Area 10.

It is required that traffic from Device A to Device D be load-balanced between Device B and Device C.

**Figure 1-281** Configuring IS-IS load balancing



| Device Name | Interface | IP Address    |
|-------------|-----------|---------------|
| Device A    | GE3/0/0   | 172.16.1.1/24 |

| Device Name | Interface | IP Address     |
|-------------|-----------|----------------|
|             | GE1/0/0   | 10.1.1.1/24    |
|             | GE2/0/0   | 10.1.2.1/24    |
| Device B    | GE1/0/0   | 10.1.1.2/24    |
|             | GE2/0/0   | 192.168.0.1/24 |
| Device C    | GE1/0/0   | 10.1.2.2/24    |
|             | GE2/0/0   | 192.168.1.1/24 |
| Device D    | GE3/0/0   | 172.17.1.1/24  |
|             | GE1/0/0   | 192.168.0.2/24 |
|             | GE2/0/0   | 192.168.1.2/24 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see [Configuring IS-IS Authentication](#). IS-IS interface authentication is used as an example. For details, see [Example for Configuring Basic IS-IS Functions](#).

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic IS-IS functions on each router for IP interworking.
2. Cancel load balancing and view the routing table.
3. Configure load balancing on Device A and view the routing table.
4. Configure load balancing on Device A.

## Data Preparation

To complete the configuration, you need the following data:

- Area addresses and levels of the four routers
- Number (1 in this example) of equal-cost routes for load balancing on Device A
- Load balancing mode on Device A

## Procedure

**Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#).

**Step 2** Configure basic IS-IS functions. For details, see [Configuration Files](#).

**Step 3** Set the maximum number of equal-cost routes for load balancing to 1 on Device A to cancel load balancing.

```
[~DeviceA] isis 1
[~DeviceA-isis-1] maximum load-balancing 1
[*DeviceA-isis-1] commit
[~DeviceA-isis-1] quit
```

# Display the routing table of Device A.

```
[~DeviceA] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-2 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 192.168.1.0/24 20 NULL GE2/0/0 10.1.2.2 A/-/-/-
 10.1.1.0/24 10 NULL GE1/0/0 Direct D/-L/-/
 172.16.1.0/24 10 NULL GE3/0/0 Direct D/-L/-/
 172.17.1.0/24 30 NULL GE1/0/0 10.1.1.2 A/-/-/-
 10.1.2.0/24 10 NULL GE2/0/0 Direct D/-L/-/
 192.168.0.0/24 20 NULL GE1/0/0 10.1.1.2 A/-/-/-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

As shown in the preceding command output, the route to 172.17.1.0 has only one next hop (10.1.1.2) after the maximum number of equal-cost routes for load balancing is set to 1. IS-IS selects the route with next hop 10.1.1.2 as the optimal route because the system ID of Device B is smaller.

**Step 4** Restore the number of equal-cost routes for load balancing on Device A to the default value.

```
[~DeviceA] isis 1
[~DeviceA-isis-1] undo maximum load-balancing
[*DeviceA-isis-1] commit
[~DeviceA-isis-1] quit
```

# Check the routing table of Device A.

```
[~DeviceA] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-2 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 192.168.1.0/24 20 NULL GE2/0/0 10.1.2.2 A/-/-/-
 10.1.1.0/24 10 NULL GE1/0/0 Direct D/-L/-/
 172.16.1.0/24 10 NULL GE3/0/0 Direct D/-L/-/
 172.17.1.0/24 30 NULL GE1/0/0 10.1.1.2 A/-/-/-
 GE2/0/0 10.1.2.2
 10.1.2.0/24 10 NULL GE2/0/0 Direct D/-L/-/
 192.168.0.0/24 20 NULL GE1/0/0 10.1.1.2 A/-/-/-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

As shown in the routing table, after the default configuration of load balancing is restored, the route with the next hop of 10.1.1.2 (Device B) and the route with the next hop of 10.1.2.2 (Device C) on routerDevice A become valid because the maximum number of equal-cost routes is 64.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
is-level level-2
network-entity 10.0000.0000.0001.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
isis enable 1

interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
isis enable 1

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
is-level level-2
network-entity 10.0000.0000.0002.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
isis enable 1

return
```

- Device C configuration file

```

sysname DeviceC

isis 1
is-level level-2
network-entity 10.0000.0000.0003.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
isis enable 1

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
isis enable 1

return
```

- Device D configuration file

```
#
```

```
sysname DeviceD
#
isis 1
 is-level level-2
 network-entity 10.0000.0000.0004.00
#
 interface GigabitEthernet1/0/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
 isis enable 1
#
 interface GigabitEthernet2/0/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
 isis enable 1
#
 interface GigabitEthernet3/0/0
 undo shutdown
 ip address 172.17.1.1 255.255.255.0
 isis enable 1
#
return
```

## Example for Configuring IS-IS to Interact with BGP

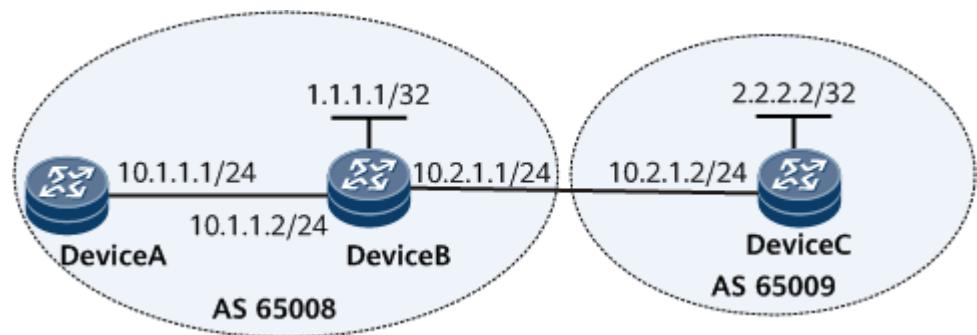
This section describes how to configure IS-IS to interact with BGP, including configuring BGP and IS-IS to import routes from each other.

### Networking Requirements

In [Figure 1-282](#):

- Device A and Device B belong to the same AS, and an IS-IS neighbor relationship is established between the two devices. BGP is not enabled on Device A.
- An EBGP connection is established between Device B and Device C. It is required that a route-policy be used to change the route cost when you configure IS-IS to import BGP routes.

**Figure 1-282** Configuring IS-IS to interact with BGP



| Device Name | Interface | IP Address     |
|-------------|-----------|----------------|
| Device A    | GE1/0/0   | 10.1.1.1/24    |
| Device B    | Loopback0 | 192.168.0.1/32 |
|             | GE1/0/0   | 10.1.1.2/24    |

| Device Name | Interface | IP Address     |
|-------------|-----------|----------------|
|             | GE2/0/0   | 10.2.1.1/24    |
| Device C    | Loopback0 | 192.168.0.2/32 |
|             | GE1/0/0   | 10.2.1.2/24    |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable IS-IS and specify NETs on Device A and Device B.
2. Establish an EBGP connection between Device B and Device C.
3. Configure IS-IS and BGP to import routes from each other on Device B, and then check the routes.

## Data Preparation

To complete the configuration, you need the following data:

- Area addresses of Device A and Device B
- Router ID and AS number of Device B
- Router ID and AS number of Device C

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic IS-IS functions.

# Configure Device A.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] commit
```

```
[~DeviceB-GigabitEthernet1/0/0] quit
```

**Step 3** Configure an EBGP connection.

```
Configure Device B.
```

```
[~DeviceB] bgp 65008
[*DeviceB-bgp] router-id 1.1.1.1
[*DeviceB-bgp] peer 10.2.1.2 as-number 65009
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] network 10.2.1.0 255.255.255.0
[*DeviceB-bgp-af-ipv4] commit
```

```
Configure Device C.
```

```
[~DeviceC] bgp 65009
[*DeviceC-bgp] router-id 2.2.2.2
[*DeviceC-bgp] peer 10.2.1.1 as-number 65008
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] network 10.2.1.0 255.255.255.0
[*DeviceC-bgp-af-ipv4] commit
```

**Step 4** Configure IS-IS to import BGP routes.

```
Configure a static route on Device C.
```

```
[~DeviceC] ip route-static 172.16.1.1 32 NULL 0
[*DeviceC] commit
```

```
On Device C, configure BGP to import the static route.
```

```
[~DeviceC] bgp 65009
[~DeviceC-bgp] import-route static
[*DeviceC-bgp] commit
```

```
On Device B, configure IS-IS to import the BGP route.
```

```
[~DeviceB] isis 1
[~DeviceB-isis-1] import-route bgp
[*DeviceB-isis-1] commit
[~DeviceB-isis-1] quit
```

```
Display the routing table of Device A. The command shows that IS-IS successfully imports the BGP route 172.16.1.1/32.
```

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table: _public_
Destinations : 11 Routes : 11
```

| Destination/Mask     | Proto          | Pre       | Cost      | Flags    | NextHop         | Interface                   |
|----------------------|----------------|-----------|-----------|----------|-----------------|-----------------------------|
| 10.1.1.0/24          | Direct         | 0         | 0         | D        | 10.1.1.1        | GigabitEthernet1/0/0        |
| 10.1.1.1/32          | Direct         | 0         | 0         | D        | 127.0.0.1       | GigabitEthernet1/0/0        |
| 10.1.1.255/32        | Direct         | 0         | 0         | D        | 10.1.1.2        | GigabitEthernet1/0/0        |
| 127.0.0.0/8          | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |
| 127.0.0.1/32         | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |
| 127.255.255.255/32   | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |
| <b>172.16.1.1/32</b> | <b>ISIS-L2</b> | <b>15</b> | <b>74</b> | <b>D</b> | <b>10.1.1.2</b> | <b>GigabitEthernet1/0/0</b> |
| 255.255.255.255/32   | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |

| Destination/Mask     | Proto          | Pre       | Cost      | Flags    | NextHop         | Interface                   |
|----------------------|----------------|-----------|-----------|----------|-----------------|-----------------------------|
| 10.1.1.0/24          | Direct         | 0         | 0         | D        | 10.1.1.1        | GigabitEthernet1/0/0        |
| 10.1.1.1/32          | Direct         | 0         | 0         | D        | 127.0.0.1       | GigabitEthernet1/0/0        |
| 10.1.1.255/32        | Direct         | 0         | 0         | D        | 10.1.1.2        | GigabitEthernet1/0/0        |
| 127.0.0.0/8          | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |
| 127.0.0.1/32         | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |
| 127.255.255.255/32   | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |
| <b>172.16.1.1/32</b> | <b>ISIS-L2</b> | <b>15</b> | <b>74</b> | <b>D</b> | <b>10.1.1.2</b> | <b>GigabitEthernet1/0/0</b> |
| 255.255.255.255/32   | Direct         | 0         | 0         | D        | 127.0.0.1       | InLoopBack0                 |

```
On Device B, configure the AS_Path filter, and apply the filter in the routing policy named RTC.
```

```
[~DeviceB] ip as-path-filter 1 permit 65009
[*DeviceB] route-policy RTC permit node 0
[*DeviceB-route-policy] if-match as-path-filter 1
[*DeviceB-route-policy] apply cost 20
```

```
[*DeviceB-route-policy] commit
[~DeviceB-route-policy] quit
```

# On Device B, configure IS-IS to import the BGP route.

```
[~DeviceB] isis 1
[~DeviceB-isis-1] import-route bgp route-policy RTC
[*DeviceB-isis-1] commit
[~DeviceB-isis-1] quit
```

# Check the routing table of Device A. The command shows that the AS\_Path filter has been applied and that the cost of the imported route 172.16.1.1/32 changes from 74 to 94.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table: _public_
Destinations : 11 Routes : 11

Destination/Mask Proto Pre Cost Flags NextHop Interface

 10.1.1.0/24 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
 10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 10.1.1.255/32 Direct 0 0 D 10.1.1.2 GigabitEthernet1/0/0
 127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
 127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
 172.16.1.1/32 ISIS-L2 15 94 D 10.1.1.2 GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

### Step 5 Configure BGP to import IS-IS routes.

```
[~DeviceB] bgp 65008
[~DeviceB-bgp] import-route isis 1
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Display the routing table of Device C. The command shows that BGP has imported the IS-IS route 10.1.1.0/24.

```
[~DeviceC] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table: _public_
Destinations : 7 Routes : 7

Destination/Mask Proto Pre Cost Flags NextHop Interface

 192.168.0.2/32 Direct 0 0 D 127.0.0.1 LoopBack0
 10.1.1.0/24 EBGP 255 0 D 10.2.1.1 GigabitEthernet1/0/0
 10.2.1.0/24 Direct 0 0 D 10.2.1.2 GigabitEthernet1/0/0
 10.2.1.2/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 10.2.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
 127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
 127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
 172.16.1.1/32 Static 60 0 DB 0.0.0.0 NULL0
 192.168.0.1/32 EBGP 255 0 RD 10.2.1.1 GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA
```

```

isis 1
network-entity 10.0000.0000.0001.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
import-route bgp route-policy RTC
network-entity 10.0000.0000.0002.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0

interface LoopBack0
ip address 192.168.0.1 255.255.255.255

bgp 65008
router-id 1.1.1.1
peer 10.2.1.2 as-number 65009

ipv4-family unicast
undo synchronization
network 10.2.1.0 255.255.255.0
import-route static
import-route isis 1
peer 10.2.1.2 enable

ip as-path-filter 1 permit 65009

route-policy RTC permit node 0
if-match as-path-filter 1
apply cost 20

return
```

- Device C configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0

interface LoopBack0
ip address 192.168.0.2 255.255.255.255

bgp 65009
router-id 2.2.2.2
peer 10.2.1.1 as-number 65008

ipv4-family unicast
undo synchronization
network 10.2.1.0 255.255.255.0
import-route static
peer 10.2.1.1 enable
```

```

ip route-static 172.16.1.1 255.255.255.255 NULL0

return
```

## Example for Configuring IS-IS Multi-Instance Processes

This section provides an example for configuring IS-IS multi-instance processes.

## Networking Requirements

On IS-IS networks, multiple IS-IS processes need to be used to isolate different access rings. To close the access rings, the IS-IS processes on all access rings need to be enabled. In this case, you need to enable different IGP processes on one interface. This reduces the interface number and configuration workload of the access rings.

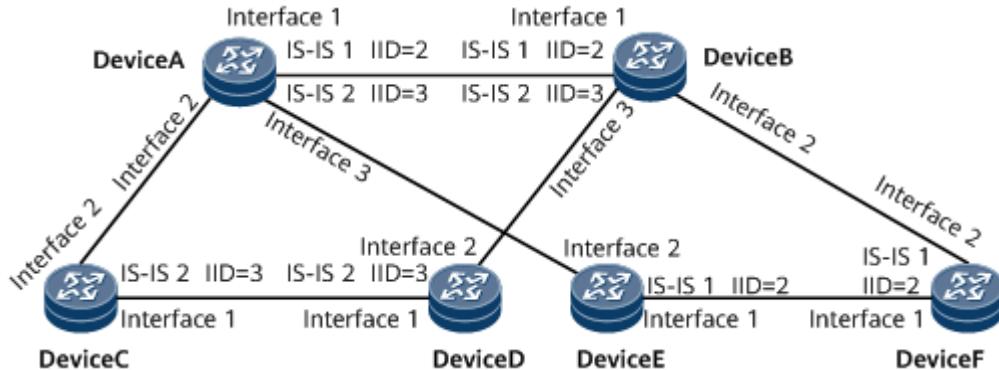
As shown in **Figure 1-283**:

Device A, Device B, Device C, Device D run IS-IS to implement IP network connectivity and form an IS-IS access ring. In addition, Device A, Device B, Device E, and Device F run IS-IS to implement IP network connectivity and form another IS-IS access ring. IS-IS multi-instance processes are enabled on all routers. Two IS-IS multi-instance processes are enabled on a specified interface of Device A and Device B, and one IS-IS multi-instance process is enabled on a specified interface of Device C, Device D, Device E, and Device F.

**Figure 1-283** Networking for configuring IS-IS multi-instance processes

 **NOTE**

Interfaces 1, 2, and 3 stand for GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



| Device Name | Interface | IP Address     |
|-------------|-----------|----------------|
| Device A    | GE 1/0/0  | 10.1.1.2/24    |
|             | GE 2/0/0  | 192.168.4.4/24 |
|             | GE 3/0/0  | 192.168.5.5/24 |
| Device B    | GE 1/0/0  | 10.1.1.1/24    |
|             | GE 2/0/0  | 192.168.0.1/24 |

| Device Name | Interface | IP Address     |
|-------------|-----------|----------------|
|             | GE 3/0/0  | 192.168.1.1/24 |
| Device C    | GE 1/0/0  | 192.168.3.1/24 |
|             | GE 2/0/0  | 192.168.4.2/24 |
| Device D    | GE 1/0/0  | 192.168.3.2/24 |
|             | GE 2/0/0  | 192.168.1.2/24 |
| Device E    | GE 1/0/0  | 192.168.2.4/24 |
|             | GE 2/0/0  | 192.168.5.4/24 |
| Device F    | GE 1/0/0  | 192.168.2.5/24 |
|             | GE 2/0/0  | 192.168.0.3/24 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IS-IS multi-instance processes on the router devices.
2. Enable two IS-IS multi-instance processes on an interface of Device A and Device B, and enable one IS-IS multi-instance process on an interface of Device C, Device D, Device E, and Device F.
3. Configure a cost for each IS-IS multi-instance process on the interface of Device A and Device B.
4. Check information about IS-IS interfaces.

## Data Preparation

To complete the configuration, you need the following data:

- Area ID of Device A, Device B, Device C, Device D, Device E, and Device F
- IS-IS multi-instance process IDs of Device A, Device B, Device C, Device D, Device E, and Device F

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

**Step 2** Configure IS-IS multi-instance processes.

```
Configure DeviceA.
```

```
[~DeviceA] isis 1
[*DeviceA-isis-1] network-entity 00.1111.1111.1111.00
[*DeviceA-isis-1] multi-instance enable iid 2
[*DeviceA-isis-1] commit
[~DeviceA-isis-1] quit
[~DeviceA] isis 2
[*DeviceA-isis-2] network-entity 00.1111.1111.1112.00
[*DeviceA-isis-2] multi-instance enable iid 3
[*DeviceA-isis-2] commit
[~DeviceA-isis-2] quit
```

# Configure DeviceB.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 00.2222.2222.2222.00
[*DeviceB-isis-1] multi-instance enable iid 2
[*DeviceB-isis-1] commit
[~DeviceB-isis-1] quit
[~DeviceB] isis 2
[*DeviceB-isis-2] network-entity 00.2222.2222.2223.00
[*DeviceB-isis-2] multi-instance enable iid 3
[*DeviceB-isis-2] commit
[~DeviceB-isis-2] quit
```

# Configure Device C, Device D, Device E, and Device F.

Repeat this step for Device C, Device D, Device E, and Device F. For configuration details, see Configuration Files in this section.

**Step 3** Configure costs on Device A and Device B.

# Configure DeviceA.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] isis enable 2
[*DeviceA-GigabitEthernet1/0/0] isis process-id 1 cost 5
[*DeviceA-GigabitEthernet1/0/0] isis process-id 2 cost 15
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure DeviceB.

```
[~DeviceB] interface gigabitethernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] isis enable 2
[*DeviceB-GigabitEthernet1/0/0] isis process-id 1 cost 5
[*DeviceB-GigabitEthernet1/0/0] isis process-id 2 cost 15
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

**Step 4** Verify the configuration.

# View information about the specified IS-IS interface on DeviceA. The command output shows that more than one multi-instance IS-IS process can be enabled on the interface.

```
[~DeviceA] display isis interface gigabitethernet 1/0/0
 Interface information for ISIS(1)

Interface Id IPV4.State IPV6.State MTU Type DIS
GE1/0/0 001 Up Mtu:Up/Lnk:Up/IP:Up 1497 L1/L2 No/No

 Interface information for ISIS(2)

Interface Id IPV4.State IPV6.State MTU Type DIS
GE1/0/0 001 Up Mtu:Up/Lnk:Dn/IP:Dn 1497 L1/L2 No/No
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
network-entity 00.1111.1111.1111.00
multi-instance enable iid 2
isis 2
network-entity 00.1111.1111.1112.00
multi-instance enable iid 3

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1
isis process-id 1 cost 5
isis enable 2
isis process-id 2 cost 15

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.4.4 255.255.255.0
isis enable 2

interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.5.5 255.255.255.0
isis enable 1

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
network-entity 00.2222.2222.2222.00
multi-instance enable iid 2
isis 2
network-entity 00.2222.2222.2223.00
multi-instance enable iid 3

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1
isis process-id 1 cost 5
isis enable 2
isis process-id 2 cost 15

interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
isis enable 1

interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
isis enable 2

return
```

- Device C configuration file

```

sysname DeviceC

isis 2
```

```
network-entity 00.3333.3333.3333.00
multi-instance enable iid 3
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.3.1 255.255.255.0
isis enable 2
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.4.2 255.255.255.0
isis enable 2
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
isis 2
network-entity 00.4444.4444.4444.00
multi-instance enable iid 3
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.3.2 255.255.255.0
isis enable 2
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
isis enable 2
#
return
```

- Device E configuration file

```
#
sysname DeviceE
#
isis 1
network-entity 00.5555.5555.5555.00
multi-instance enable iid 2
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.2.4 255.255.255.0
isis enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.5.4 255.255.255.0
isis enable 1
#
return
```

- Device F configuration file

```
#
sysname DeviceF
#
isis 1
network-entity 00.6666.6666.6666.00
multi-instance enable iid 2
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.2.5 255.255.255.0
isis enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
```

```
ip address 192.168.0.3 255.255.255.0
isis enable 1
#
return
```

## Example for Configuring Static BFD for IS-IS

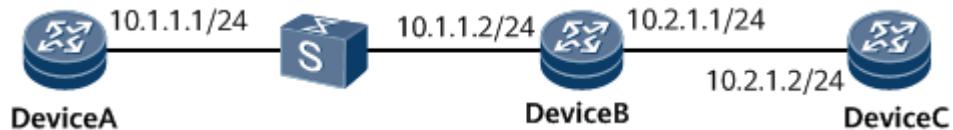
This section describes how to configure static BFD for IS-IS, including configuring BFD parameters and enabling static BFD.

## Networking Requirements

In [Figure 1-284](#):

- Device A and Device B are connected through a Layer 2 switch.
- IS-IS runs on Device A, Device B, and Device C.
- BFD is configured to detect the IS-IS neighbor relationship between Device A and Device B. If the link between Device A and Device B fails, BFD can quickly detect the fault and notify it to IS-IS.

**Figure 1-284** Configuring static BFD for IS-IS



| Device Name | Interface | IP Address  |
|-------------|-----------|-------------|
| Device A    | GE1/0/0   | 10.1.1.1/24 |
| Device B    | GE1/0/0   | 10.1.1.2/24 |
|             | GE2/0/0   | 10.2.1.1/24 |
| Device C    | GE1/0/0   | 10.2.1.2/24 |

### NOTE

BFD for IS-IS cannot be used to detect the multi-hop link between Device A and Device C because the IS-IS neighbor relationship is not established between Device A and Device C.

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable basic IS-IS functions on each router.

## 2. Enable BFD on Device A and Device B.

### Data Preparation

To complete the configuration, you need the following data:

- IS-IS process ID
- Area addresses of Device A, Device B, and Device C
- Levels of Device A, Device B, and Device C
- Name of the BFD session established between Device A and Device B and peer IP address to be detected by BFD
- Local and remote discriminators of the BFD session established between Device A and Device B

### Procedure

**Step 1** Configure an IP address for the interface on each router. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic IS-IS functions.

# Configure Device A.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-2
[*DeviceA-isis-1] network-entity aa.1111.1111.1111.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] is-level level-2
[*DeviceB-isis-1] network-entity aa.2222.2222.2222.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] isis enable 1
[*DeviceB-GigabitEthernet2/0/0] commit
[~DeviceB-GigabitEthernet2/0/0] quit
```

# Configure Device C.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] is-level level-2
[*DeviceC-isis-1] network-entity aa.3333.3333.3333.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] commit
[~DeviceC-GigabitEthernet1/0/0] quit
```

# After the preceding configurations. The command shows that the neighbor relationship has been established between Device A and Device B.

```
[~DeviceA] display isis peer
```

| Peer information for ISIS(1) |           |                   |       |          |      |     |
|------------------------------|-----------|-------------------|-------|----------|------|-----|
| System Id                    | Interface | Circuit Id        | State | HoldTime | Type | PRI |
| 2222.2222.2222               | GE1/0/0   | 2222.2222.2222.00 | Up    | 23s      | L2   | 64  |
| Total Peer(s): 1             |           |                   |       |          |      |     |

The IS-IS routing table of Device A has the routes to Device B and Device C.

| Route information for ISIS(1)    |         |         |               |          |                                                                                             |  |
|----------------------------------|---------|---------|---------------|----------|---------------------------------------------------------------------------------------------|--|
| ISIS(1) Level-2 Forwarding Table |         |         |               |          |                                                                                             |  |
| IPV4 Destination                 | IntCost | ExtCost | ExitInterface | NextHop  | Flags                                                                                       |  |
| 10.1.1.0/24                      | 10      | NULL    | GE1/0/0       | Direct   | D/-L/-                                                                                      |  |
| 10.2.1.0/24                      | 20      | NULL    | GE1/0/0       | 10.1.1.2 | A/-/-                                                                                       |  |
|                                  |         |         |               |          | Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,<br>U-Up/Down Bit Set |  |
|                                  |         |         |               |          | Protect Type: L-Link Protect, N-Node Protect                                                |  |

### Step 3 Configure BFD.

# Enable BFD and configure a BFD session on Device A.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA-bfd] bfd atob bind peer-ip 10.1.1.2 interface gigabitethernet1/0/0
[*DeviceA-bfd-session-atob] discriminator local 1
[*DeviceA-bfd-session-atob] discriminator remote 2
[*DeviceA-bfd-session-atob] commit
[~DeviceA-bfd-session-atob] quit
```

# Enable BFD and configure a BFD session on Device B.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB-bfd] bfd btoa bind peer-ip 10.1.1.1 interface gigabitethernet1/0/0
[*DeviceB-bfd-session-btoa] discriminator local 2
[*DeviceB-bfd-session-btoa] discriminator remote 1
[*DeviceB-bfd-session-btoa] commit
[~DeviceB-bfd-session-btoa] quit
```

# After the preceding configurations, run the **display bfd session** command on Device A or Device B. The command shows that the BFD session has been Up.

Use the command output on Device A as an example.

```
[~DeviceA] display bfd session all
(w): State in WTR
(*) State is invalid

Local Remote PeerIpAddr State Type Interface Name

1 2 10.1.1.2 Up S_IP_IF GigabitEthernet1/0/0

Total UP/DOWN Session Number : 1/0
```

### Step 4 Enable static BFD on the IS-IS interface.

# Configure Device A.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] isis bfd static
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] quit
```

# Configure Device B.

```
[~DeviceB] interface gigabitethernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] isis bfd static
[*DeviceB-GigabitEthernet1/0/0] commit
```

**Step 5** Verify the configuration.

# Enable debugging on Device A.

```
<DeviceA> debugging isis adjacency 1 interface GigabitEthernet1/0/0
<DeviceA> debugging isis circuit-information 1 interface GigabitEthernet1/0/0
<DeviceA> terminal debugging
<DeviceA> terminal monitor
```

# Run the **shutdown** command on GigabitEthernet 1/0/0 of Device B to simulate a link fault.

```
[~DeviceB-GigabitEthernet1/0/0] shutdown
[*DeviceB-GigabitEthernet1/0/0] commit
```

# Display the logs on Device A. The command output shows that IS-IS has deleted the neighbor relationship between Device A and Device B after BFD notifies the fault.

```
#80/active/IsisAdjacencyChange/Major/occurredTime:2011-03-09 04:17:07/-/-/alarm1
D:0x08960007/VS=0:ISIS adjacency state change. (SysInstance=1, SysLevel=1, Circl
ndex=2, CircIndex=20, LspId=2222.2222.2222.00.00, AdjState=1, IfIndex=20, IfNa
me=GE1/0/0, Reason=BFD detected that the neighbor went Down, SubReason=14)
```

----End

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
bfd
#
isis 1
is-level level-2
network-entity aa.1111.1111.1111.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1
isis bfd static
#
bfd atob bind peer-ip 10.1.1.2 interface GigabitEthernet1/0/0
discriminator local 1
discriminator remote 2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
bfd
#
isis 1
is-level level-2
network-entity aa.2222.2222.2222.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
```

```
isis enable 1
isis bfd static
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0
isis enable 1
#
bfd btoa bind peer-ip 10.1.1.1 interface GigabitEthernet1/0/0
discriminator local 2
discriminator remote 1
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
isis 1
is-level level-2
network-entity aa.3333.3333.3333.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0
isis enable 1
#
return
```

## Example for Configuring Dynamic BFD for IS-IS

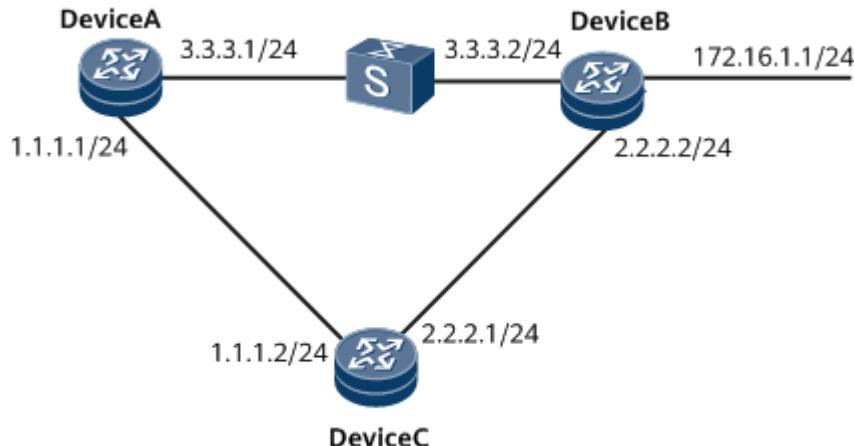
This section describes how to configure dynamic BFD for IS-IS, including configuring basic IS-IS functions, setting the interface cost, configuring BFD for the IS-IS process, and configuring BFD for the IS-IS interface on each device.

## Networking Requirements

As shown in [Figure 1-285](#):

- IS-IS runs on Device A, Device B, and Device C.
- BFD is enabled for the IS-IS processes on Device A, Device B, and Device C.
- Service traffic is transmitted along the primary link Device A -> Device B. The link Device A -> Device C -> Device B functions as the backup link.
- BFD is configured for the interfaces on the link between Device A and Device B. If the link fails, BFD can fast detect the fault and notify IS-IS of the fault so that service traffic can be transmitted through the backup link.

Figure 1-285 Configuring dynamic BFD for IS-IS



| Device Name | Interface | IP Address    |
|-------------|-----------|---------------|
| Device A    | GE1/0/0   | 1.1.1.1/24    |
|             | GE2/0/0   | 3.3.3.1/24    |
| Device B    | GE1/0/0   | 2.2.2.2/24    |
|             | GE2/0/0   | 3.3.3.2/24    |
|             | GE3/0/0   | 172.16.1.1/24 |
| Device C    | GE1/0/0   | 1.1.1.2/24    |
|             | GE2/0/0   | 2.2.2.1/24    |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic IS-IS functions on each router for IP interworking.
2. Set the IS-IS interface cost to control route selection.
3. Enable global BFD.
4. Enable BFD for the IS-IS process on Device A, Device B, and Device C.
5. Enable BFD for the interfaces on Device A and Device B.

## Data Preparation

To complete the configuration, you need the following data:

- IS-IS process ID
- Areas of Device A, Device B, and Device C
- Interface costs of Device A and Device B
- Numbers and types of the interfaces to be enabled with BFD on Device A, Device B, and Device C
- Minimum interval at which BFD packets are received and sent and local detection multiplier on Device A and Device B

## Procedure

**Step 1** Configure an IP address for the interface on each router. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic IS-IS functions.

# Configure Device A.

```
[~DeviceA] isis
[*DeviceA-isis-1] is-level level-2
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] isis enable 1
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis
[*DeviceB-isis-1] is-level level-2
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] isis enable 1
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface gigabitethernet 3/0/0
[*DeviceB-GigabitEthernet3/0/0] isis enable 1
[*DeviceB-GigabitEthernet3/0/0] commit
[~DeviceB-GigabitEthernet3/0/0] quit
```

# Configure Device C.

```
[~DeviceC] isis
[*DeviceC-isis-1] is-level level-2
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] isis enable 1
[*DeviceC-GigabitEthernet2/0/0] commit
[~DeviceC-GigabitEthernet2/0/0] quit
```

# Run the **display isis peer** command. The command output shows that the neighbor relationships have been established between Device A and Device B, and between Device A and Device C. Device A is used as an example.

```
[~DeviceA] display isis peer
```

|                  |         |                   |    |     |    |    |
|------------------|---------|-------------------|----|-----|----|----|
| 0000.0000.0002   | GE2/0/0 | 0000.0000.0002.01 | Up | 9s  | L2 | 64 |
| 0000.0000.0003   | GE1/0/0 | 0000.0000.0001.02 | Up | 21s | L2 | 64 |
| Total Peer(s): 2 |         |                   |    |     |    |    |

# The routers have learned routes from each other. The following uses the routing table of Device A as an example.

```
[~DeviceA] display ip routing-table
```

Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

```
Routing Table: _public_
```

| Destinations : 8     | Routes : 9  |           |           |          |                |                             |
|----------------------|-------------|-----------|-----------|----------|----------------|-----------------------------|
| Destination/Mask     | Proto       | Pre       | Cost      | Flags    | NextHop        | Interface                   |
| 1.1.1.0/24           | Direct      | 0         | 0         | D        | 1.1.1.1        | GigabitEthernet1/0/0        |
| 1.1.1.1/32           | Direct      | 0         | 0         | D        | 127.0.0.1      | InLoopBack0                 |
| 2.2.2.0/24           | ISIS        | 15        | 20        | D        | 1.1.1.2        | GigabitEthernet1/0/0        |
| 3.3.3.0/24           | Direct      | 0         | 0         | D        | 3.3.3.1        | GigabitEthernet2/0/0        |
| 3.3.3.1/32           | Direct      | 0         | 0         | D        | 127.0.0.1      | InLoopBack0                 |
| 127.0.0.0/8          | Direct      | 0         | 0         | D        | 127.0.0.1      | InLoopBack0                 |
| 127.0.0.1/32         | Direct      | 0         | 0         | D        | 127.0.0.1      | InLoopBack0                 |
| <b>172.16.1.0/24</b> | <b>ISIS</b> | <b>15</b> | <b>20</b> | <b>D</b> | <b>3.3.3.2</b> | <b>GigabitEthernet2/0/0</b> |

As shown in the routing table, the next hop address of the route to 172.16.1.0/24 is 3.3.3.2, and traffic is transmitted on the primary link Device A -> Device B.

### Step 3 Set the interface cost.

```
Configure Device A.
```

```
[~DeviceA] interface gigabitethernet 2/0/0
[~DeviceA-GigabitEthernet2/0/0] isis cost 5
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

```
Configure Device B.
```

```
[~DeviceB] interface gigabitethernet 2/0/0
[~DeviceB-GigabitEthernet2/0/0] isis cost 5
[*DeviceB-GigabitEthernet2/0/0] commit
[~DeviceB-GigabitEthernet2/0/0] quit
```

### Step 4 Configure BFD for the IS-IS process.

```
Enable BFD for the IS-IS process on Device A.
```

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] isis
[*DeviceA-isis-1] bfd all-interfaces enable
[*DeviceA-isis-1] commit
[~DeviceA-isis-1] quit
```

```
Enable BFD for the IS-IS process on Device B.
```

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] isis
[*DeviceB-isis-1] bfd all-interfaces enable
[*DeviceB-isis-1] commit
[~DeviceB-isis-1] quit
```

```
Enable BFD for the IS-IS process on Device C.
```

```
[~DeviceC] bfd
[*DeviceC-bfd] quit
[*DeviceC] isis
[*DeviceC-isis-1] bfd all-interfaces enable
```

```
[*DeviceC-isis-1] commit
[~DeviceC-isis-1] quit
```

# Run the **display isis bfd session all** command on Device A, Device B, or Device C. The command output shows that the BFD session is up.

Use the command output on Device A as an example.

```
[~DeviceA] display isis bfd session all
 BFD session information for ISIS(1)

Peer System ID : 0000.0000.0002 Interface : GE2/0/0
TX : 10 BFD State : up Peer IP Address : 3.3.3.2
RX : 10 LocDis : 16385 Local IP Address: 3.3.3.1
Multiplier : 3 RemDis : 16388 Type : L2
Diag : No diagnostic information

Peer System ID : 0000.0000.0003 Interface : GE1/0/0
TX : 10 BFD State : up Peer IP Address : 1.1.1.2
RX : 10 LocDis : 16386 Local IP Address: 1.1.1.1
Multiplier : 3 RemDis : 16387 Type : L2
Diag : No diagnostic information

Total BFD session(s): 2
```

The preceding information shows that the BFD sessions between Device A and Device B and between Device A and Device C are Up.

#### Step 5 Configure BFD for IS-IS interfaces.

# On GE 2/0/0 of Device A, configure BFD and set the minimum interval at which BFD packets are received and sent to 100 ms and the local detection multiplier to 4.

```
[~DeviceA] interface gigabitEthernet 2/0/0
[~DeviceA-GigabitEthernet2/0/0] isis bfd enable
[*DeviceA-GigabitEthernet2/0/0] isis bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

# On GE 2/0/0 of Device B, configure BFD and set the minimum interval at which BFD packets are received and sent to 100 ms and the local detection multiplier to 4.

```
[~DeviceB] interface gigabitethernet 2/0/0
[~DeviceB-GigabitEthernet2/0/0] isis bfd enable
[*DeviceB-GigabitEthernet2/0/0] isis bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceB-GigabitEthernet2/0/0] commit
[~DeviceB-GigabitEthernet2/0/0] quit
```

# Run the **display isis bfd session all** command on Device A or Device B. The command outputs show that BFD parameters have already taken effect. Use the command output on Device B as an example.

```
[~DeviceB] display isis bfd session all
 BFD session information for ISIS(1)

Peer System ID : 0000.0000.0001 Interface : GE2/0/0
TX : 100 BFD State : up Peer IP Address : 3.3.3.1
RX : 100 LocDis : 16385 Local IP Address: 3.3.3.2
Multiplier : 4 RemDis : 16385 Type : L2
Diag : No diagnostic information

Peer System ID : 0000.0000.0003 Interface : GE1/0/0
TX : 10 BFD State : up Peer IP Address : 2.2.2.1
RX : 10 LocDis : 16385 Local IP Address: 2.2.2.2
Multiplier : 4 RemDis : 16385 Type : L2
```

Diag : No diagnostic information

Total BFD session(s): 2

**Step 6** # Run the **shutdown** command on Gigabit Ethernet 2/0/0 on Device B to simulate the fault on the primary link.

```
[~DeviceB] interface gigabitethernet 2/0/0
[~DeviceB-GigabitEthernet2/0/0] shutdown
[*DeviceB-GigabitEthernet2/0/0] commit
```

**Step 7** Verify the configuration.

# Display the routing table of Device A.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Destinations : 9 Routes : 9
Destination/Mask Proto Pre Cost Flags NextHop Interface
1.1.1.0/24 Direct 0 0 D 1.1.1.1 GigabitEthernet1/0/0
1.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
1.1.1.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
2.2.2.0/24 ISIS 15 20 D 1.1.1.2 GigabitEthernet1/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
172.16.1.0/24 ISIS 15 30 D 1.1.1.2 GigabitEthernet1/0/0
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
```

As shown in the routing table, the backup link Device A -> Device C -> Device B takes effect after the primary link fails, and the next hop address of the route to 172.16.1.0/24 becomes 1.1.1.2.

# Run the **display isis bfd session all** command on Device A. The command output shows that only the BFD session between Device A and Device C is up.

```
[~DeviceA] display isis bfd session all
BFD session information for ISIS(1)

Peer System ID : 0000.0000.0003 Interface : GE1/0/0
TX : 10 BFD State : up Peer IP Address : 1.1.1.2
RX : 10 LocDis : 16385 Local IP Address: 1.1.1.1
Multiplier : 3 RemDis : 16388 Type : L2
Diag : No diagnostic information
```

Total BFD session(s): 1

----End

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
bfd
#
isis 1
is-level level-2
bfd all-interfaces enable
network-entity 10.0000.0000.0001.00
#
interface GigabitEthernet1/0/0
```

```
undo shutdown
ip address 1.1.1.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 3.3.3.1 255.255.255.0
isis enable 1
isis cost 5
isis bfd enable
isis bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
#
return
```

- Device B configuration file

```

sysname DeviceB

bfd

isis 1
is-level level-2
bfd all-interfaces enable
network-entity 10.0000.0000.0002.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 2.2.2.2 255.255.255.0
isis enable 1

interface GigabitEthernet2/0/0
undo shutdown
ip address 3.3.3.2 255.255.255.0
isis enable 1
isis cost 5
isis bfd enable
isis bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
isis enable 1
#
return
```

- Device C configuration file

```

sysname DeviceC

bfd

isis 1
is-level level-2
bfd all-interfaces enable
network-entity 10.0000.0000.0003.00

interface GigabitEthernet1/0/0
undo shutdown
ip address 1.1.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet2/0/0
undo shutdown
ip address 2.2.2.1 255.255.255.0
isis enable 1
#
return
```

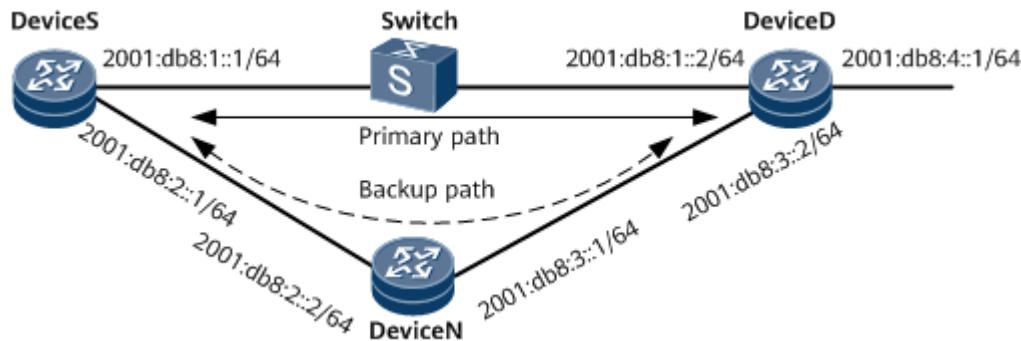
## Example for Configuring Dynamic BFD for IPv6 IS-IS

This section provides an example for configuring dynamic BFD to fast detect failures and trigger fast switchover of service traffic on IS-IS IPv6 networks.

## Networking Requirements

**Figure 1-286** shows an IS-IS IPv6 network. The primary link between Device S and Device D is Device S -> Switch -> Device D; the backup link between Device S and Device D is Device S -> Device N -> Device D.

**Figure 1-286** Configuring dynamic BFD for IPv6 IS-IS



| Device Name | Interface | IP Address       |
|-------------|-----------|------------------|
| Device S    | GE1/0/0   | 2001:db8:1::1/64 |
|             | GE2/0/0   | 2001:db8:2::1/64 |
| Device N    | GE1/0/0   | 2001:db8:2::2/64 |
|             | GE2/0/0   | 2001:db8:3::1/64 |
| Device D    | GE1/0/0   | 2001:db8:1::2/64 |
|             | GE2/0/0   | 2001:db8:3::2/64 |
|             | GE3/0/0   | 2001:db8:4::1/64 |

It is required to configure BFD for IPv6 IS-IS so that traffic between Device S and Device D can be rapidly switched to the backup path if the primary link or Switch fails.

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic IS-IS IPv6 functions on each router to ensure that IPv6 routes are reachable.
2. Adjust the IS-IS cost on each router interface so that the link of Device S -> Switch -> Device D functions as the primary link and the link of Device S -> Device N -> Device D functions as the backup link.
3. Enable BFD globally on each router.
4. Enable BFD for IPv6 IS-IS in the IS-IS view of each router.

## Data Preparation

To complete the configuration, you need the following data:

- IS-IS process ID
- IS-IS NET
- Level of each router
- IS-IS cost of each interface
- Numbers of interfaces to be enabled with BFD for IPv6 IS-IS
- Minimum interval at which BFD packets are received and sent, and BFD local detection time multiplier

## Procedure

### Step 1 Enable IPv6 and configure IPv6 addresses for all interfaces.

# Use the configuration of Device S as an example. The configuration of any other router is similar to that of Device S.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceS
[*HUAWEI] commit
[~DeviceS] interface gigabitethernet 1/0/0
[~DeviceS-GigabitEthernet1/0/0] ipv6 enable
[*DeviceS-GigabitEthernet1/0/0] ipv6 address 2001:db8:1::1/64
[*DeviceS-GigabitEthernet1/0/0] commit
[~DeviceS-GigabitEthernet1/0/0] quit
```

### Step 2 Configure basic IS-IS IPv6 functions.

# Configure Device S.

```
[~DeviceS] isis 10
[*DeviceS-isis-10] is-level level-2
[*DeviceS-isis-10] network-entity 10.0000.0000.0001.00
[*DeviceS-isis-10] ipv6 enable
[*DeviceS-isis-10] quit
[*DeviceS] interface gigabitethernet 1/0/0
[*DeviceS-GigabitEthernet1/0/0] isis ipv6 enable 10
[*DeviceS-GigabitEthernet1/0/0] quit
[*DeviceS] interface gigabitethernet 2/0/0
[*DeviceS-GigabitEthernet2/0/0] isis ipv6 enable 10
[*DeviceS-GigabitEthernet2/0/0] commit
[~DeviceS-GigabitEthernet2/0/0] quit
```

# Configure Device N.

```
[~DeviceN] isis 10
[*DeviceN-isis-10] is-level level-2
[*DeviceN-isis-10] network-entity 10.0000.0000.0002.00
[*DeviceN-isis-10] ipv6 enable
```

```
[*DeviceN-isis-10] quit
[*DeviceN] interface gigabitethernet 1/0/0
[*DeviceN-GigabitEthernet1/0/0] isis ipv6 enable 10
[*DeviceN-GigabitEthernet1/0/0] quit
[*DeviceN] interface gigabitethernet 2/0/0
[*DeviceN-GigabitEthernet2/0/0] isis ipv6 enable 10
[*DeviceN-GigabitEthernet2/0/0] commit
[~DeviceN-GigabitEthernet2/0/0] quit
```

# Configure Device D.

```
[~DeviceD] isis 10
[*DeviceD-isis-10] is-level level-2
[*DeviceD-isis-10] network-entity 10.0000.0000.0003.00
[*DeviceD-isis-10] ipv6 enable
[*DeviceD-isis-10] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] isis ipv6 enable 10
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis ipv6 enable 10
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] interface gigabitethernet 3/0/0
[*DeviceD-GigabitEthernet3/0/0] isis ipv6 enable 10
[*DeviceD-GigabitEthernet3/0/0] commit
[~DeviceD-GigabitEthernet3/0/0] quit
```

# Run the **display ipv6 routing-table** command. The command output shows that the routers have learned IPv6 routes from each other.

**Step 3** Configure the IS-IS cost for each interface.

# Configure Device S.

```
[~DeviceS] interface gigabitethernet 1/0/0
[~DeviceS-GigabitEthernet1/0/0] isis ipv6 cost 1 level-2
[*DeviceS-GigabitEthernet1/0/0] quit
[*DeviceS] interface gigabitethernet 2/0/0
[*DeviceS-GigabitEthernet2/0/0] isis ipv6 cost 10 level-2
[*DeviceS-GigabitEthernet2/0/0] commit
[~DeviceS-GigabitEthernet2/0/0] quit
```

# Configure Device N.

```
[~DeviceN] interface gigabitethernet 1/0/0
[~DeviceN-GigabitEthernet1/0/0] isis ipv6 cost 10 level-2
[*DeviceN-GigabitEthernet1/0/0] quit
[*DeviceN] interface gigabitethernet 2/0/0
[*DeviceN-GigabitEthernet2/0/0] isis ipv6 cost 10 level-2
[*DeviceN-GigabitEthernet2/0/0] commit
[~DeviceN-GigabitEthernet2/0/0] quit
```

# Configure Device D.

```
[~DeviceD] interface gigabitethernet 1/0/0
[~DeviceD-GigabitEthernet1/0/0] isis ipv6 cost 1 level-2
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis ipv6 cost 10 level-2
[*DeviceD-GigabitEthernet2/0/0] commit
[~DeviceD-GigabitEthernet2/0/0] quit
```

**Step 4** Configure BFD for IPv6 IS-IS.

# Enable BFD for IPv6 IS-IS globally on Device S, Device N, and Device D, set the minimum interval at which BFD packets are received and sent to 150 ms, and set the local detection time multiplier to 3 (default value).

# Configure Device S.

```
[~DeviceS] bfd
[*DeviceS-bfd] quit
[*DeviceS] isis 10
[*DeviceS-isis-10] ipv6 bfd all-interfaces enable
[*DeviceS-isis-10] ipv6 bfd all-interfaces min-tx-interval 150 min-rx-interval 150
[*DeviceS-isis-10] commit
[~DeviceS-isis-10] quit
```

# Configure Device N.

```
[~DeviceN] bfd
[*DeviceN-bfd] quit
[*DeviceN] isis 10
[*DeviceN-isis-10] ipv6 bfd all-interfaces enable
[*DeviceN-isis-10] ipv6 bfd all-interfaces min-tx-interval 150 min-rx-interval 150
[*DeviceN-isis-10] commit
[~DeviceN-isis-10] quit
```

# Configure Device D.

```
[~DeviceD] bfd
[*DeviceD-bfd] quit
[*DeviceD] isis 10
[*DeviceD-isis-10] ipv6 bfd all-interfaces enable
[*DeviceD-isis-10] ipv6 bfd all-interfaces min-tx-interval 150 min-rx-interval 150
[*DeviceD-isis-10] commit
[~DeviceD-isis-10] quit
```

# Run the **display isis ipv6 bfd session all** command on Device S or Device D. The command outputs show that the BFD parameters have taken effect. Use the command output on Device S as an example.

```
[~DeviceS] display isis ipv6 bfd session all

```

```
IPv6 BFD session information for ISIS(10)

Peer System ID : 0000.0000.0003 Type : L2
Interface : GE1/0/0
IPv6 BFD State : up TX : 150 RX : 150 Multiplier : 3
LocDis : 16386 Local IPv6 Address: FE80::E0:2F47:B103:1
RemDis : 16386 Peer IPv6 Address : FE80::E0:2F47:B107:1
Diag : No diagnostic information

Peer System ID : 0000.0000.0002 Type : L2
Interface : GE2/0/0
IPv6 BFD State : up TX : 150 RX : 150 Multiplier : 3
LocDis : 16386 Local IPv6 Address: FE80::C964:0:B203:1
RemDis : 16386 Peer IPv6 Address : FE80::C964:0:B8B6:1
Diag : No diagnostic information
```

Total BFD session(s): 2

### Step 5 Verify the configuration.

# Run the **display ipv6 routing-table 2001:db8:4::1 64** command on Device S to check the IPv6 routing table. The next hop address is FE80::E0:2F47:B107:1; the outbound interface is GE 1/0/0.

```
[~DeviceS] display ipv6 routing-table 2001:db8:4::1 64
```

```
Routing Table : public
Summary Count : 1
```

```
Destination : 2001:db8:4:: PrefixLength : 64
NextHop : FE80::E0:2F47:B107:1 Preference : 15
Cost : 11 Protocol : ISIS
RelayNextHop : :: TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : D
```

# Run the **shutdown** command on GE 1/0/0 of Device D to simulate a fault in the primary link.

```
[~DeviceD] interface gigabitethernet 1/0/0
[~DeviceD-GigabitEthernet1/0/0] shutdown
[*DeviceD-GigabitEthernet1/0/0] commit
```

```
Run the display ipv6 routing-table 2001:db8:4::1 64 command on Device S to
check the IPv6 routing table.
```

```
[~DeviceS] display ipv6 routing-table 2001:db8:4::1 64
Routing Table : public
Summary Count : 1
```

|                                  |                   |
|----------------------------------|-------------------|
| Destination : 2001:db8:4::       | PrefixLength : 64 |
| NextHop : FE80::C964:0:B8B6:1    | Preference : 15   |
| Cost : 20                        | Protocol : ISIS   |
| RelayNextHop ::                  | TunnelID : 0x0    |
| Interface : GigabitEthernet2/0/0 | Flags : D         |

Based on the routing table, after the primary link fails, the backup link takes effect. The next hop address of the route to 2001:db8:4::/64 changes to FE80::C964:0:B8B6:1; the outbound interface changes to GE 2/0/0; the route cost also changes.

# Run the **display isis ipv6 bfd session all** command on Device S. The command output shows that only one BFD session is Up between Device S and Device N.

[~DeviceS] **display isis ipv6 bfd 10 session all**  
IPv6 BFD session information for ISIS(10)

Peer System ID : 0000.0000.0002 Type : L2  
Interface : GE2/0/0  
IPv6 BFD State : up TX : 150 RX : 150 Multiplier : 3  
LocDis : 16386 Local IPv6 Address: FE80::C964:0B20:3:1  
RemDis : 16386 Peer IPv6 Address : FE80::C964:0:B8B6:1  
Diag : No diagnostic information

Total BFD session(s): 1

----End

## Configuration Files

- Device S configuration file

```

sysname DeviceS

bfd

isis 10
is-level level-2
network-entity 10.0000.0000.0001.00

ipv6 enable topology ipv6
ipv6 bfd all-interfaces enable
ipv6 bfd all-interfaces min-tx-interval 150 min-rx-interval 150

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
isis ipv6 enable 10
```

```
isis ipv6 cost 1 level-2
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
isis ipv6 enable 10
isis ipv6 cost 10 level-2
#
return
```

- Device N configuration file

```
#
sysname DeviceN
#
bfd
#
isis 10
is-level level-2
network-entity 10.0000.0000.0002.00
#
ipv6 enable topology ipv6
ipv6 bfd all-interfaces enable
ipv6 bfd all-interfaces min-tx-interval 150 min-rx-interval 150
#
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
isis ipv6 enable 10
isis ipv6 cost 10 level-2
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
isis ipv6 enable 10
isis ipv6 cost 10 level-2
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
bfd
#
isis 10
is-level level-2
network-entity 10.0000.0000.0003.00
#
ipv6 enable topology ipv6
ipv6 bfd all-interfaces enable
ipv6 bfd all-interfaces min-tx-interval 150 min-rx-interval 150
#
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
isis ipv6 enable 10
isis ipv6 cost 1 level-2
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::2/64
isis ipv6 enable 10
isis ipv6 cost 10 level-2
```

```

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:4::1/64
isis ipv6 enable 10

return
```

- Switch configuration file:  
The configuration is not provided here.

## Example for Configuring Basic IS-IS IPv6 Functions

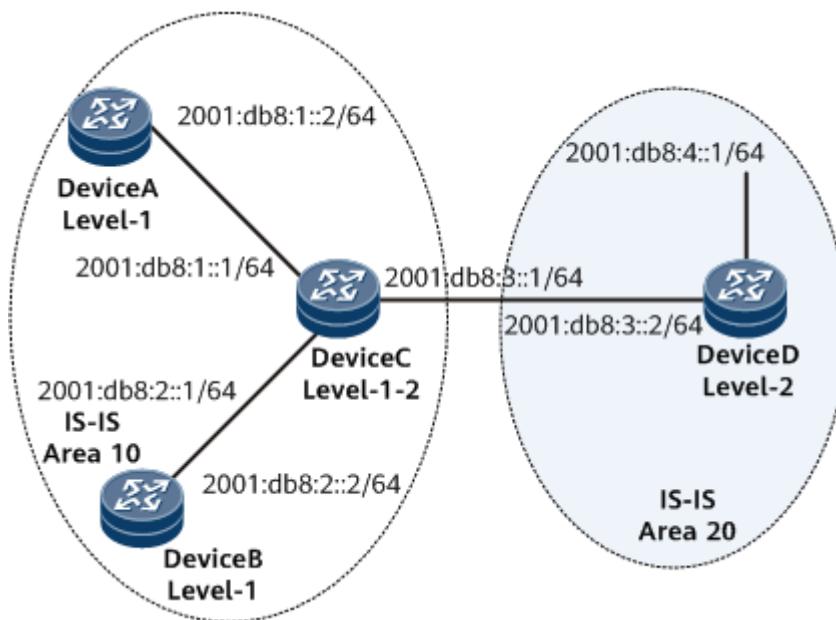
This section describes how to configure basic IS-IS IPv6 functions, including enabling IPv6 globally, configuring an IPv6 address and enabling IPv6 for each interface, and configuring basic IS-IS functions and enabling IPv6.

## Networking Requirements

As shown in [Figure 1-287](#):

- Device A, Device B, Device C, and Device D belong to the same AS. It is required that IS-IS run on them to implement IPv6 interworking.
- Device A, Device B, and Device C belong to Area 10, and Device D belongs to Area 20.
- Device A and Device B are Level-1 devices; Device C is a Level-1-2 device; Device D is a Level-2 device.

**Figure 1-287** Configuring basic IS-IS IPv6 functions



| Device Name | Interface | IP Address       |
|-------------|-----------|------------------|
| Device A    | GE 1/0/0  | 2001:db8:1::2/64 |
| Device B    | GE 1/0/0  | 2001:db8:2::2/64 |

| Device Name | Interface | IP Address       |
|-------------|-----------|------------------|
| Device C    | GE 1/0/0  | 2001:db8:1::1/64 |
|             | GE2/0/0   | 2001:db8:2::1/64 |
|             | GE 3/0/0  | 2001:db8:3::1/64 |
| Device D    | GE 1/0/0  | 2001:db8:3::2/64 |
|             | GE 2/0/0  | 2001:db8:4::1/64 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable the IPv6 forwarding capability on each router, and configure an IPv6 address for each interface.
2. Enable IS-IS, configure the level, and specify the NET on each router.

## Data Preparation

To complete the configuration, you need the following data:

- IPv6 addresses of the interfaces on Device A, Device B, Device C, and Device D
- Areas of Device A, Device B, Device C, and Device D
- Levels of Device A, Device B, Device C, and Device D

## Procedure

- Step 1** Enable IPv6, and configure an IPv6 address for each interface. Use the command output on Device A as an example. The configurations of the other three routers are the same as that of Device A. For configuration details, see [Configuration Files](#) in this section.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
[*HUAWEI] commit
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ipv6 enable
[*DeviceA-GigabitEthernet1/0/0] ipv6 address 2001:db8:1::2 64
[*DeviceA-GigabitEthernet1/0/0] commit
```

- Step 2** Configure IS-IS.

```
Configure Device A.

[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-1
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
```

```
[*DeviceA-isis-1] ipv6 enable topology ipv6
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] is-level level-1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] ipv6 enable topology ipv6
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

# Configure Device C.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] ipv6 enable topology ipv6
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] interface gigabitethernet 3/0/0
[*DeviceC-GigabitEthernet3/0/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet3/0/0] isis circuit-level level-1-2
[*DeviceC-GigabitEthernet3/0/0] commit
[~DeviceC-GigabitEthernet3/0/0] quit
```

# Configure Device D.

```
[~DeviceD] isis 1
[*DeviceD-isis-1] is-level level-2
[*DeviceD-isis-1] network-entity 20.0000.0000.0004.00
[*DeviceD-isis-1] ipv6 enable topology ipv6
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis ipv6 enable 1
[*DeviceD-GigabitEthernet2/0/0] commit
[~DeviceD-GigabitEthernet2/0/0] quit
```

### Step 3 Verify the configuration.

# Display the IS-IS routing table of Device A. The command output shows that Device A has the routes to each network segment of the Level-1 area.

```
[~DeviceA] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV6 Dest. ExitInterface NextHop Cost Flags

 ::/0 GigabitEthernet1/0/0 FE80::A83E:0:3ED2:1 10 A/-/
 2001:db8:1::/64 GigabitEthernet1/0/0 Direct 10 D/L/-
 2001:db8:2::/64 GigabitEthernet1/0/0 FE80::A83E:0:3ED2:1 20 A/-/
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

# Display detailed information about IS-IS neighbors on Device C.

```
[~DeviceC] display isis peer verbose
 Peer information for ISIS(1)
 System Id Interface Circuit Id State HoldTime Type PRI

0000.0000.0001 GigabitEthernet1/0/0 0000000001 Up 24s L1 --
 MT IDs supported : 0(UP)
 Local MT IDs : 0
 Area Address(es) : 10
 Peer IPv6 Address(es): FE80::996B:0:9419:1
 Peer IPv6 GlbAddr(es): 2001:db8:1::2/64
 Uptime : 00h00m15s
 Peer Up Time : 2020-06-08 01:41:57
 Adj Protocol : IPV6
 Restart Capable : YES
 Suppressed Adj : NO
 Peer System ID : 0000.0000.0001
 BFD Incr-Cost State : MT0 : NO / MT2 : NO
0000.0000.0002 GigabitEthernet2/0/0 0000000001 Up 28s L1 --
 Local MT IDs : 0
 Area Address(es) : 10
 Peer IPv6 Address(es): FE80::DC40:0:47A9:1
 Peer IPv6 GlbAddr(es): 2001:db8:2::2/64
 Uptime : 00h00m15s
 Peer Up Time : 2020-06-08 01:41:57
 Adj Protocol : IPV6
 Restart Capable : YES
 Suppressed Adj : NO
 Peer System ID : 0000.0000.0002
 BFD Incr-Cost State : MT0 : NO / MT2 : NO
0000.0000.0004 GigabitEthernet3/0/0 0000000001 Up 24s L2 --
 Local MT IDs : 0
 Area Address(es) : 20
 Peer IPv6 Address(es): FE80::F81D:0:1E24:2
 Peer IPv6 GlbAddr(es): 2001:db8:3::2/64
 Uptime : 00h00m15s
 Peer Up Time : 2020-06-08 01:41:57
 Adj Protocol : IPV6
 Restart Capable : YES
 Suppressed Adj : NO
 Peer System ID : 0000.0000.0004
 BFD Incr-Cost State : MT0 : NO / MT2 : NO
Total Peer (s): 3
```

# Display detailed information about the IS-IS LSDB of Device C.

```
[~DeviceC] display isis lsdb verbose
 Database information for ISIS(1)

 Level-1 Link State Database
 LSPID Seq Num Checksum Holdtime Length ATT/P/OI

0000.0000.0001.00-00 0x0000000c 0x4e06 1117 113 0/0/0
 SOURCE 0000.0000.0001.00
 NLPID IPV6
 AREA ADDR 10
 INTF ADDR V6 2001:db8:1::2
 Topology Standard
 NBR ID 0000.0000.0003.00 COST: 10
 IPV6 2001:db8:1::/64 COST: 10
0000.0000.0002.00-00 0x00000009 0x738c 1022 83 0/0/0
 SOURCE 0000.0000.0002.00
 NLPID IPV6
 AREA ADDR 10
 INTF ADDR V6 2001:db8:2::2
 Topology Standard
 NBR ID 0000.0000.0003.00 COST: 10
 IPV6 2001:db8:2::/64 COST: 10
0000.0000.0003.00-00* 0x00000020 0x6b10 771 140 1/0/0
```

```
SOURCE 0000.0000.0003.00
NLPID IPV6
AREA ADDR 10
INTF ADDR V6 2001:db8:3::1
INTF ADDR V6 2001:db8:2::1
INTF ADDR V6 2001:db8:1::1
Topology Standard
NBR ID 0000.0000.0002.00 COST: 10
NBR ID 0000.0000.0001.00 COST: 10
IPV6 2001:db8:2::/64 COST: 10
IPV6 2001:db8:1::/64 COST: 10
Total LSP (s): 5
 *(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
 ATT-Attached, P-Partition, OL-Overload
 Level-2 Link State Database
LSPID Seq Num Checksum Holdtime Length ATT/P/OL

0000.0000.0003.00-00* 0x000000017 0x61b4 771 157 0/0/0
SOURCE 0000.0000.0003.00
NLPID IPV6
AREA ADDR 10
INTF ADDR V6 2001:db8:3::1
INTF ADDR V6 2001:db8:2::1
INTF ADDR V6 2001:db8:1::1
Topology Standard
NBR ID 0000.0000.0004.00 COST: 10
IPV6 2001:db8:3::/64 COST: 10
IPV6 2001:db8:2::/64 COST: 10
IPV6 2001:db8:1::/64 COST: 10
0000.0000.0004.00-00 0x0000000b 0x6dfa 1024 124 0/0/0
SOURCE 0000.0000.0004.00
NLPID IPV6
AREA ADDR 20
INTF ADDR V6 2001:db8:3::2
INTF ADDR V6 2001:db8:4::1
Topology Standard
NBR ID 0000.0000.0003.00 COST: 10
NBR ID 0000.0000.0005.00 COST: 10
IPV6 2001:db8:3::/64 COST: 10
IPV6 2001:db8:4::/64 COST: 10
Total LSP(s): 3
 *(In TLV)-Leaking Route, *(By LSPID)-Self LSP, +-Self LSP(Extended),
 ATT-Attached, P-Partition, OL-Overload
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
 is-level level-1
 ipv6 enable topology ipv6
 network-entity 10.0000.0000.0001.00

interface GigabitEthernet1/0/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 isis ipv6 enable 1

return
```

- Device B configuration file

```

sysname DeviceB
```

```

isis 1
is-level level-1
ipv6 enable topology ipv6
network-entity 10.0000.0000.0002.00

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
isis ipv6 enable 1

return
```

- Device C configuration file

```

sysname DeviceC

isis 1
ipv6 enable topology ipv6
network-entity 10.0000.0000.0003.00

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
isis ipv6 enable 1

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
isis ipv6 enable 1

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::1/64
isis ipv6 enable 1
isis circuit-level level-1-2

return
```

- Device D configuration file

```

sysname DeviceD

isis 1
is-level level-2
ipv6 enable topology ipv6
network-entity 20.0000.0000.0004.00

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:3::2/64
isis ipv6 enable 1

interface GigabitEthernet2/0/0
ipv6 enable
ipv6 address 2001:db8:4::1/64
isis ipv6 enable 1

return
```

## Example for Configuring IS-IS Auto FRR

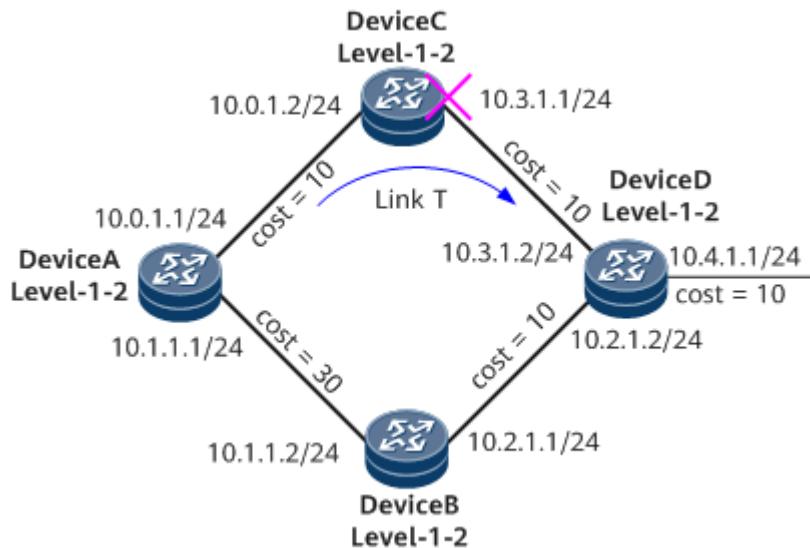
### Networking Requirements

If a fault occurs on a network, IS-IS Auto FRR fast switches traffic to a backup link before the route convergence, which prevents traffic interruption.

In [Figure 1-288](#):

- IS-IS runs between four routers.
- The four routers are all Level-1-2 routers.
- If DeviceC or Link T fails, it is required that the traffic forwarded by DeviceA be rapidly switched to the backup link.

**Figure 1-288** Configuring IS-IS Auto FRR



| Device Name | Interface | IP Address  |
|-------------|-----------|-------------|
| DeviceA     | GE1/0/0   | 10.0.1.1/24 |
|             | GE2/0/0   | 10.1.1.1/24 |
| DeviceB     | GE1/0/0   | 10.1.1.2/24 |
|             | GE2/0/0   | 10.2.1.1/24 |
| DeviceC     | GE1/0/0   | 10.0.1.2/24 |
|             | GE2/0/0   | 10.3.1.1/24 |
| DeviceD     | GE1/0/0   | 10.3.1.2/24 |
|             | GE2/0/0   | 10.2.1.2/24 |
|             | GE3/0/0   | 10.4.1.1/24 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable basic IS-IS functions on each router.
2. Set a higher link cost (in compliance with the traffic protection inequality of IS-IS Auto FRR) on GE 2/0/0 of DeviceA, and ensure that Link T is preferentially selected.
3. Enable IS-IS Auto FRR on DeviceA that forwards the protected traffic.

## Data Preparation

To complete the configuration, you need the following data:

- IP addresses of interfaces on each router
- NET of each router
- Level of each router
- Costs of interfaces on each router

## Procedure

**Step 1** Configure IP addresses for interfaces. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic IS-IS functions.

# Configure DeviceA.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-1-2
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] isis enable 1
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

# Configure DeviceB.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] is-level level-1-2
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] isis enable 1
[*DeviceB-GigabitEthernet2/0/0] commit
[~DeviceB-GigabitEthernet2/0/0] quit
```

# Configure DeviceC.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] is-level level-1-2
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] isis enable 1
[*DeviceC-GigabitEthernet2/0/0] commit
[~DeviceC-GigabitEthernet2/0/0] quit
```

# Configure DeviceD.

```
[~DeviceD] isis 1
[*DeviceD-isis-1] is-level level-1-2
[*DeviceD-isis-1] network-entity 10.0000.0000.0004.00
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] isis enable 1
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis enable 1
[*DeviceD-GigabitEthernet2/0/0] commit
[~DeviceD-GigabitEthernet2/0/0] quit
```

**Step 3** Set the cost of Gigabit Ethernet 2/0/0 on Device A to 30, and then check routing information.

# Configure the cost of GE 2/0/0 on DeviceA to 30.

```
[~DeviceA] interface gigabitethernet 2/0/0
[~DeviceA-GigabitEthernet2/0/0] isis cost 30
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

# Check information about the link from DeviceA to DeviceD. IS-IS preferentially selects link T to transmit traffic forwarded by DeviceA because link T has the smallest cost.

```
[~DeviceA] display isis route verbose 10.4.1.1
```

Route information for ISIS(1)

-----

ISIS(1) Level-1 Forwarding Table

```
IPV4 Dest : 10.4.1.0/24 Int. Cost : 30 Ext. Cost : NULL
Admin Tag : - Src Count : 2 Flags : A-/L-
Priority : -
NextHop : Interface : ExitIndex :
 10.0.1.2 GE1/0/0 0x00000010
Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
Protect Type: L-Link Protect, N-Node Protect
```

-----

ISIS(1) Level-2 Forwarding Table

```
IPV4 Dest : 10.4.1.0/24 Int. Cost : 30 Ext. Cost : NULL
Admin Tag : - Src Count : 2 Flags : -/-/-/-
Priority : -
NextHop : Interface : ExitIndex :
 -
Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
```

U-Up/Down Bit Set  
Protect Type: L-Link Protect, N-Node Protect

**Step 4** Enable IS-IS Auto FRR.

```
Enable IS-IS Auto FRR on DeviceA.
```

```
[~DeviceA] isis
[~DeviceA-isis-1] frr
[*DeviceA-isis-1-frr] loop-free-alternate
[*DeviceA-isis-1-frr] commit
[~DeviceA-isis-1-frr] quit
[~DeviceA-isis-1] quit
```

**Step 5** Verify the configuration.

```
Check information about the route from DeviceA to DeviceD. The command
output shows that IS-IS generates a backup route because IS-IS Auto FRR is
enabled.
```

```
[~DeviceA] display isis route verbose 10.4.1.1
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV4 Dest : 10.4.1.0/24 Int. Cost : 30 Ext. Cost : NULL
 Admin Tag : - Src Count : 2 Flags : A-/L-
 Priority : -
 NextHop : Interface : ExitIndex :
 10.0.1.2 GE1/0/0 0x00000010
 (B)10.1.1.2 GE2/0/0 0x0000000f(N)
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect

 ISIS(1) Level-2 Forwarding Table

 IPV4 Dest : 10.4.1.0/24 Int. Cost : 30 Ext. Cost : NULL
 Admin Tag : - Src Count : 2 Flags : -/-/-/-
 Priority : -
 NextHop : Interface : ExitIndex :
 -
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
isis 1
network-entity 10.0000.0000.0001.00
frr
loop-free-alternate level-1
loop-free-alternate level-2
#
interface GigabitEthernet 1/0/0
undo shutdown
ip address 10.0.1.1 255.255.255.0
isis enable 1
```

```

interface GigabitEthernet 2/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
isis enable 1
isis cost 30

return
```

- DeviceB configuration file

```

sysname DeviceB

isis 1
network-entity 10.0000.0000.0002.00

interface GigabitEthernet 1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet 2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0
isis enable 1

return
```

- DeviceC configuration file

```

sysname DeviceC

isis 1
network-entity 10.0000.0000.0003.00

interface GigabitEthernet 1/0/0
undo shutdown
ip address 10.0.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet 2/0/0
undo shutdown
ip address 10.3.1.1 255.255.255.0
isis enable 1

return
```

- DeviceD configuration file

```

sysname DeviceD

isis 1
network-entity 10.0000.0000.0004.00

interface GigabitEthernet 1/0/0
undo shutdown
ip address 10.3.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet 2/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet 3/0/0
undo shutdown
ip address 10.4.1.1 255.255.255.0
isis enable 1

return
```

## Example for Configuring IS-IS MT

The following example shows how to implement network interconnection using IS-IS on a network with an IPv4/IPv6 topology.

## Networking Requirements

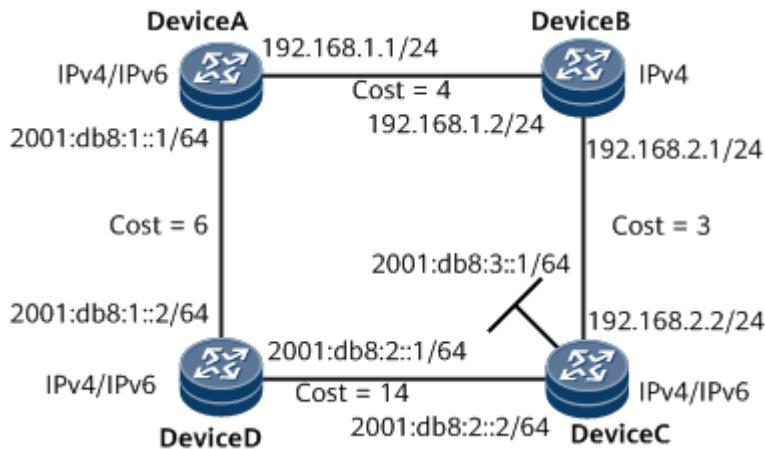
If an IPv4/IPv6 topology is deployed on a network, various end-to-end services, such as voice and data services share the same physical links. As a result, IPv4 or IPv6 packets are discarded, affecting transmission quality. To address this issue, establish a separate routing table for IPv6 using multi-topology.

In [Figure 1-289](#), Device A, Device C, and Device D support both IPv4 and IPv6, whereas Device B supports only IPv4.

Loopback1 on DeviceC needs to be reachable. If IS-IS MT is not supported, the shortest path calculated using SPF passes through DeviceB, which does not support IPv6. As a result, IPv6 packets cannot reach the destination.

To ensure IPv6 packet transmission, enable IS-IS MT and create separate IPv4 and IPv6 routing tables.

**Figure 1-289** Configuring IS-IS MT



| Device Name | Interface | IP Address       |
|-------------|-----------|------------------|
| Device A    | GE 1/0/0  | 192.168.1.1/24   |
|             | GE 2/0/0  | 2001:db8:1::1/64 |
| Device B    | GE 1/0/0  | 192.168.1.2/24   |
|             | GE 2/0/0  | 192.168.2.1/24   |
| Device C    | Loopbak1  | 2001:db8:3::1/64 |
|             | GE 1/0/0  | 2001:db8:2::1/64 |
|             | GE 2/0/0  | 192.168.2.2/24   |
| Device D    | GE 1/0/0  | 2001:db8:1::2/64 |

| Device Name | Interface | IP Address       |
|-------------|-----------|------------------|
|             | GE 2/0/0  | 2001:db8:2::1/64 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IPv4/IPv6 addresses for the interfaces on routers so that devices in different areas can communicate.
2. Enable IPv6 as well as global IPv4 and IPv6 topologies on the routers supporting the IPv4/IPv6 dual stack and enable a global IPv4 topology on Device B.
3. Configure basic IS-IS functions and link costs.
4. Create separate IPv4 and IPv6 topology instances on the routers supporting the IPv4/IPv6 dual stack and create an IPv4 topology instance on router B.
5. Associate the interfaces with specified topology instances.

## Data Preparation

To complete the configuration, you need the following data:

- IP addresses of the interfaces on routers as shown in [Figure 1-289](#), area ID (86), system ID of Device A (0000.0000.0001), system ID of Device B (0000.0000.0002), system ID of Device C (0000.0000.0003), system ID of Device D (0000.0000.0004), and level of all the routers (Level-1)
- Cost of the link from Device D to Device A (6), cost of the link from Device A to Device B (4), cost of the link from Device B to Device C (3), cost of the link from Device D to Device C (14)
- IPv4 topology instance **red** for all routers and IPv6 topology instance **blue** for Device A, Device C, and Device D

## Procedure

### Step 1 Configure IP addresses for interfaces.

Configure an IPv4 address and a mask for each interface on Device B, and configure IPv4 as well as IPv6 addresses and masks for each interface on Device A, Device C, and Device D based on [Figure 1-289](#). For configuration details, see [Configuration Files](#) in this section.

### Step 2 Enable IPv6 as well as global IPv4 and IPv6 topologies on the routers supporting the IPv4/IPv6 dual stack and enable a global IPv4 topology on Device B.

# Enable global IPv4 and IPv6 topologies on router A.

```
[~DeviceA] ip topology red
[*DeviceA] ipv6 topology blue
[*DeviceA] commit
```

The configurations on Device C and Device D are similar to that on router A. For configuration details, see [Configuration Files](#) in this section.

# Enable the global IPv4 topology on Device B.

```
[~DeviceB] ip topology red
[*DeviceB] commit
```

**Step 3** Configure basic IS-IS functions and link costs.

For details about the configuration of basic IS-IS functions, see [Examples for Configuring Basic IS-IS Functions](#).

# Set the cost of the link from Device A to Device B to 4.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] isis cost 4
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

The configuration of other link costs is similar to that of the link from Device A to Device B.

**Step 4** Create an IPv4 topology instance **red** for each router and an IPv6 topology instance **blue** for Device A, Device C, and Device D.

# Associate an IS-IS process with the IPv4 topology instance **red** and IPv6 topology instance **blue** on Device A.

```
[~DeviceA] isis
[~DeviceA-isis-1] ipv6 enable topology ipv6
[*DeviceA-isis-1] cost-style wide
[*DeviceA-isis-1] topology red topology-id 10
[*DeviceA-isis-1-topology-red] commit
[~DeviceA-isis-1-topology-red] quit
[*DeviceA-isis-1] ipv6 topology blue topology-id 20
[*DeviceA-isis-1-topology-blue] commit
[~DeviceA-isis-1-topology-blue] quit
[~DeviceA-isis-1] quit
```

The configurations on Device C and Device D are similar to that on Device A.

# Associate an IS-IS process with the IPv4 topology instance **red** on Device B.

```
[~DeviceB] isis
[~DeviceB-isis-1] cost-style wide
[*DeviceB-isis-1] topology red topology-id 10
[*DeviceB-isis-1-topology-red] commit
[~DeviceB-isis-1-topology-red] quit
[~DeviceB-isis-1] quit
```

**Step 5** Associate the interfaces with specified topology instances.

# Use the interfaces on Device A as an example.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ip topology red enable
[*DeviceA-GigabitEthernet1/0/0] isis topology red
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] interface gigabitethernet 2/0/0
[~DeviceA-GigabitEthernet2/0/0] ip topology red enable
[*DeviceA-GigabitEthernet2/0/0] isis topology red
[*DeviceA-GigabitEthernet2/0/0] ipv6 topology blue enable
```

```
[*DeviceA-GigabitEthernet2/0/0] isis ipv6 topology blue
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

**Step 6** Verify the configuration.

Run the **display isis route** command on the routers to view information about the learned routes. The command output on Device D is used as an example.

# Display the routing information on Device D.

```
[~DeviceD] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV6 Dest. ExitInterface NextHop Cost Flags

 2001:db8:3::/64 GE2/0/0 FE80::D11:0:36D4:1 14 A/-/
 2001:db8:2::/64 GE2/0/0 Direct 14 D/L-
 2001:db8:1::/64 GE1/0/0 Direct 6 D/L-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

IPv6 routes are calculated only on the IPv6 topology. Therefore, the outbound interface of the route from Device D to 2001:db8:3::/64 is GE 2/0/0.

# Run the **tracert** command on Device D.

```
[~DeviceD] tracert ipv6 2001:db8:3::1
traceroute to 2001:db8:3::1 30 hops max,60 bytes packet
1 2001:db8:3::1 62 ms 63 ms 31 ms
```

To make a comparison between the preceding routing information and the routing information when an IPv4/IPv6 topology is deployed, run the following commands.

```
[~DeviceD] isis 1
[~DeviceD-isis-1] ipv6 enable
[*DeviceD-isis-1] commit
```

Configuration modifications on Device A and Device C are similar to that on Device D.

After the configuration is modified, run the **display isis route** command on the routers once again to view information about the learned routes. Use the command output on DeviceD as an example.

# Display the routing information on Device D.

```
[~DeviceD] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV6 Dest. ExitInterface NextHop Cost Flags

 2001:db8:3::/64 GE1/0/0 FE80::200:5EFF:FE01:100 13 A/-/
 2001:db8:2::/64 GE2/0/0 Direct 14 D/L-
 2001:db8:1::/64 GE1/0/0 Direct 6 D/L-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

The preceding output shows that the outbound interface of the route from Device D to 2001:db8:3::/64 is GE1/0/0 because in integrated topology calculation, the cost of the link from this interface to 2001:db8:3::1/64 is smaller.

```
[~DeviceD] tracert ipv6 2001:db8:3::1
traceroute to 2001:db8:3::1 30 hops max,60 bytes packet
1 2001:db8:1::1 31 ms !N 31 ms !N 32 ms !N
```

However, the **tracert** command output shows that IPv6 packets cannot reach the destination.

```
Display the routing information on Device A.
```

```
[~DeviceA] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 192.168.2.0/24 7 NULL GE1/0/0 192.168.1.2 A/-/-/
 192.168.1.0/24 4 NULL GE1/0/0 Direct D/-L-
 IPV6 Dest. ExitInterface NextHop Cost Flags

 2001:db8:2::/64 GE1/0/0 FE80::2E0:A9FF:FE47:8302 24 A/-
 2001:db8:1::/64 GE2/0/0 Direct 10 D/L-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
 Protect Type: L-Link Protect, N-Node Protect
```

The preceding output shows that there is no outbound interface for the route from Device A to 2001:db8:3::/64 because the link between Device A and Device B does not support IPv6 and IPv6 packets from Device D are discarded.

----End

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
ip topology red
#
ipv6 topology blue
#
isis 1
cost-style wide
network-entity 86.0000.0000.0001.00
ipv6 enable topology ipv6
#
topology red topology-id 10
#
ipv6 topology blue topology-id 20
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
isis enable 1
isis cost 4
ip topology red enable
isis topology red
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
```

```
ipv6 address 2001:db8:1::1/64
isis ipv6 enable 1
ipv6 topology blue enable
isis ipv6 topology blue
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
ip topology red
#
isis 1
network-entity 86.0000.0000.0002.00
#
topology red topology-id 10
#
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
isis enable 1
ip topology red enable
isis topology red
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.1 255.255.255.0
isis enable 1
isis cost 3
ip topology red enable
isis topology red
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
ip topology red
#
ipv6 topology blue
#
isis 1
cost-style wide
network-entity 86.0000.0000.0003.00
ipv6 enable topology ipv6
#
topology red topology-id 10
#
ipv6 topology blue topology-id 20
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64
isis ipv6 enable 1
ip topology red enable
isis topology red
ipv6 topology blue enable
isis ipv6 topology blue
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0
isis enable 1
ip topology red enable
isis topology red
#
```

```
interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:3::1/64
isis ipv6 enable 1
#
return
```

- Device D configuration file

```

sysname DeviceD

ipv6 topology blue

isis 1
is-level level-1
cost-style wide
network-entity 86.0000.0000.0004.00
ipv6 enable topology ipv6

ipv6 topology blue topology-id 20

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64
isis ipv6 enable 1
isis cost 6
isis ipv6 cost 6
ipv6 topology blue enable
isis ipv6 topology blue

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64
isis ipv6 enable 1
isis cost 14
isis ipv6 cost 14
ipv6 topology blue enable
isis ipv6 topology blue

return
```

## Example for Configuring Routing Loop Detection for Routes Imported from OSPF to IS-IS

This section provides an example for configuring routing loop detection for routes imported from OSPF to IS-IS.

### Networking Requirements

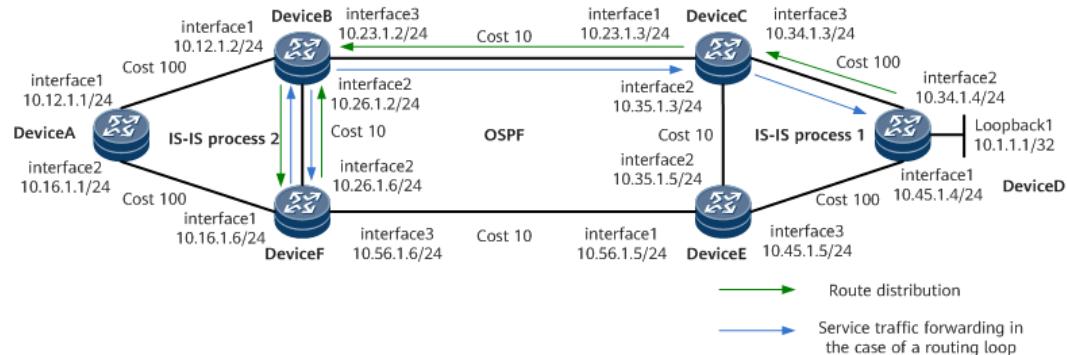
On the live network, IS-IS routes can be imported to an OSPF process for redistribution. In such a scenario, routing policies are usually configured on multiple devices to prevent routing loops. If routing policies are incorrectly configured on the devices that import routes, routing loops may occur. To prevent this problem, configure routing loop detection for the routes imported to IS-IS.

On the network shown in [Figure 1-290](#), IS-IS process 2 is configured on DeviceA, DeviceB, and DeviceF, an OSPF process is configured on DeviceB, DeviceC, DeviceE, and DeviceF, and IS-IS process 1 is configured on DeviceC, DeviceD, and DeviceE. DeviceC is configured to import routes from IS-IS process 1 to the OSPF process, DeviceB is configured to import routes from the OSPF process to IS-IS process 2, and DeviceF is configured to import routes from IS-IS process 2 to the OSPF process.

**Figure 1-290** Routing loop detection for routes imported from OSPF to IS-IS

 NOTE

In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IP addresses for interfaces on each device to implement area connectivity.
2. Enable IS-IS and OSPF, and configure basic IS-IS and OSPF functions.
3. Configure route import to construct a routing loop.
4. View routing entries to check whether a routing loop occurs.
5. Enable routing loop detection and view routing entries to check whether the routing loop is eliminated.

## Procedure

**Step 1** Configure an IP address for each interface.

DeviceA, DeviceB, DeviceC, and DeviceD are used as examples.

# Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ip address 10.12.1.1 24
[*DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] ip address 10.16.1.1 24
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] interface gigabitethernet 1/0/0
```

```
[*DeviceB-GigabitEthernet1/0/0] ip address 10.12.1.2 24
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] ip address 10.26.1.2 24
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface gigabitethernet 3/0/0
[*DeviceB-GigabitEthernet3/0/0] ip address 10.23.1.2 24
[*DeviceB-GigabitEthernet3/0/0] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
<DeviceC> system-view
[~DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] ip address 10.23.1.3 24
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ip address 10.35.1.3 24
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] interface gigabitethernet 3/0/0
[*DeviceC-GigabitEthernet3/0/0] ip address 10.34.1.3 24
[*DeviceC-GigabitEthernet3/0/0] quit
[*DeviceC] commit
```

# Configure DeviceD. The route 10.1.1.1/32 on the loopback interface of DeviceD is used as the locally originated route.

```
<DeviceD> system-view
[~DeviceD] interface loopBack 1
[*DeviceD-LoopBack1] ip address 10.1.1.1 32
[*DeviceD-LoopBack1] isis enable 1
[*DeviceD-LoopBack1] quit
[*DeviceD-LoopBack1] commit
```

The configurations of other devices are similar to those of the preceding devices. For details, see [Configuration Files](#) in this section.

**Step 2** Enable IS-IS and OSPF, and configure basic IS-IS and OSPF functions and a link cost to implement intra-area interworking.

# Configure IS-IS process 2 on DeviceA, DeviceB, and DeviceF. DeviceA is used as an example.

```
[~DeviceA] isis 2
[*DeviceA-isis-2] cost-style wide
[*DeviceA-isis-2] network-entity 10.1111.0000.0001.00
[*DeviceA-isis-2] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis enable 2
[*DeviceA-GigabitEthernet1/0/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet1/0/0] isis cost 10
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] isis enable 2
[*DeviceA-GigabitEthernet2/0/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet2/0/0] isis cost 10
[*DeviceA-GigabitEthernet2/0/0] quit
[*DeviceA] commit
```

The configurations of DeviceB and DeviceF are similar to those of DeviceA. For configuration details, see [Configuration Files](#) in this section.

# Configure IS-IS process 1 on DeviceC, DeviceD, and DeviceE. DeviceD is used as an example.

```
[~DeviceD] isis 1
[*DeviceD-isis-1] cost-style wide
[*DeviceD-isis-1] network-entity 10.4444.0000.0001.00
```

```
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] isis enable 1
[*DeviceD-GigabitEthernet1/0/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet1/0/0] isis cost 10
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis enable 1
[*DeviceD-GigabitEthernet2/0/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet2/0/0] isis cost 10
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] commit
```

The configurations of DeviceC and DeviceE are similar to those of DeviceD. For configuration details, see [Configuration Files](#) in this section.

# Configure OSPF on DeviceB, DeviceC, DeviceE, and DeviceF. DeviceB is used as an example.

```
[~DeviceB] ospf 1 router-id 2.2.2.2
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] ospf enable 1 area 0.0.0.0
[*DeviceB-GigabitEthernet1/0/0] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] ospf enable 1 area 0.0.0.0
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] commit
```

The configurations of DeviceC, DeviceD, and DeviceE are similar to those of DeviceB. For configuration details, see [Configuration Files](#) in this section.

**Step 3** Configure route import.

# Configure DeviceC to import routes from IS-IS process 1 to the OSPF process.

```
[~DeviceC] ospf 1 router-id 3.3.3.3
[*DeviceC-ospf-1] import-route isis 1 cost 100 type 2
[*DeviceC-ospf-1] commit
```

# Configure DeviceB to import routes from the OSPF process to IS-IS process 2.

```
[~DeviceB] isis 2
[*DeviceB-isis-2] import-route ospf 1
[*DeviceB-ospf-1] commit
```

# Configure DeviceF to import routes from IS-IS process 2 to the OSPF process.

```
[~DeviceF] ospf 1 router-id 6.6.6.6
[*DeviceF-ospf-1] import-route isis 2 cost 10 type 2
[*DeviceF-ospf-1] commit
```

**Step 4** View the routing tables on DeviceB, DeviceE, and DeviceF to check whether a routing loop occurs.

# Check OSPF neighbor information on DeviceB.

```
[~DeviceB] display ospf peer
(M) Indicates MADJ neighbor

OSPF Process 1 with Router ID 2.2.2.2
Neighbors

Area 0.0.0.0 interface 10.23.1.2 (GigabitEthernet3/0/0)'s neighbors
Router ID: 3.3.3.3 Address: 10.23.1.3
State: Full Mode:Nbr is Master Priority: 1
```

```
DR: 10.23.1.3 BDR: 10.23.1.2 MTU: 0
Dead timer due in 3 sec
Retrans timer interval: 5
Neighbor is up for 28h43m46s
Neighbor Up Time : 2021-08-23 09:20:09
Authentication Sequence: [0]

Area 0.0.0.0 interface 10.26.1.2 (GigabitEthernet2/0/0)'s neighbors
Router ID: 6.6.6.6 Address: 10.26.1.6
State: Full Mode:Nbr is Master Priority: 1
DR: None BDR: None MTU: 0
Dead timer due in 4 sec
Retrans timer interval: 5
Neighbor is up for 28h43m46s
Neighbor Up Time : 2021-08-23 09:20:09
Authentication Sequence: [0]
```

# Display IS-IS neighbor information on Device B.

```
[~DeviceB] display isis peer
Peer information for ISIS(2)

System Id Interface Circuit Id State HoldTime Type PRI

6666.0000.0001 GE3/0/0 0000000011 Up 29s L1L2 --
1111.0000.0001 GE2/0/0 0000000025 Up 27s L1L2 --

Total Peer(s): 2
```

The preceding command outputs show that DeviceB has established connections with DeviceA, DeviceC, and DeviceF.

# Check the routing table on DeviceB.

```
[~DeviceB] display ip routing 10.1.1.1
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.1/24 O_ASE 150 1 D 10.26.1.6 GigabitEthernet3/0/0
```

The preceding command output shows that the next hop of the route displayed on DeviceB is DeviceF.

# Check OSPF neighbor information on DeviceF.

```
[~DeviceF] display ospf peer
(M) Indicates MADJ neighbor

OSPF Process 1 with Router ID 6.6.6.6
Neighbors

Area 0.0.0.0 interface 10.56.1.6 (GigabitEthernet3/0/0)'s neighbors
Router ID: 5.5.5.5 Address: 10.56.1.5
State: Full Mode:Nbr is Slave Priority: 1
DR: 10.56.1.6 BDR: 10.56.1.5 MTU: 0
Dead timer due in 4 sec
Retrans timer interval: 5
Neighbor is up for 28h52m49s
Neighbor Up Time : 2021-08-23 09:20:11
Authentication Sequence: [0]

Area 0.0.0.0 interface 10.26.1.6 (GigabitEthernet2/0/0)'s neighbors
Router ID: 2.2.2.2 Address: 10.26.1.2
State: Full Mode:Nbr is Slave Priority: 1
```

```
DR: None BDR: None MTU: 0
Dead timer due in 3 sec
Retrans timer interval: 5
Neighbor is up for 28h52m51s
Neighbor Up Time : 2021-08-23 09:20:09
Authentication Sequence: [0]
```

# Check IS-IS neighbor information on DeviceF.

```
[~DeviceF] display isis peer
Peer information for ISIS(2)

System Id Interface Circuit Id State HoldTime Type PRI

2222.0000.0001 GE2/0/0 0000000011 Up 28s L1L2 --
1111.0000.0001 GE1/0/0 0000000011 Up 25s L1L2 --
Total Peer(s): 2
```

The preceding command outputs show that DeviceF has established connections with DeviceA, DeviceB, and DeviceE.

# Check the routing table on DeviceF.

```
[~DeviceF] display ip routing 10.1.1.1
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.1/24 ISIS-L2 15 10 D 10.26.1.2 GigabitEthernet2/0/0
```

The preceding command output shows that the next hop of the route displayed on DeviceF is DeviceB.

In this case, a routing loop is formed between DeviceB and DeviceF.

**Step 5** Enable routing loop detection on each device.

# Configure DeviceA, which is used as an example.

```
[*DeviceA] route loop-detect isis enable
[*DeviceA] route loop-detect ospf enable
[*DeviceA] commit
```

The configurations of other devices are similar to those of DeviceA. For configuration details, see [Configuration Files](#) in this section.

**Step 6** Check whether the routing loop is eliminated.

# Check the routing table on DeviceB.

```
[~DeviceB] display ip routing 10.1.1.1
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.1/24 O_ASE 150 100 D 10.23.1.3 GigabitEthernet3/0/0
```

The preceding command output shows that the next hop of the route displayed on DeviceB is DeviceC.

# Check the routing table on DeviceF.

```
[~DeviceF] display ip routing 10.1.1.1
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination/Mask Proto Pre Cost Flags NextHop Interface
10.1.1.1/24 ISIS-L2 15 10 D 10.26.1.2 GigabitEthernet2/0/0
```

The preceding command output shows that the next hop of the route displayed on DeviceF is DeviceB and that DeviceB no longer prefers the route received from DeviceF. This means that the routing loop between DeviceB and DeviceF is eliminated.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

isis 2
cost-style wide
network-entity 10.1111.0000.0001.00

route loop-detect isis enable

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.12.1.1 255.255.255.0
isis enable 2
isis circuit-type p2p
isis cost 10

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.16.1.1 255.255.255.0
isis enable 2
isis circuit-type p2p
isis cost 10

route loop-detect ospf enable

return
```

- DeviceB configuration file

```

sysname DeviceB

isis 2
cost-style wide
network-entity 10.2222.0000.0001.00
import-route ospf 1

route loop-detect isis enable

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.12.1.2 255.255.255.0
isis enable 2
isis circuit-type p2p
isis cost 10

interface GigabitEthernet2/0/0
undo shutdown
```

```
ip address 10.26.1.2 255.255.255.0
ospf network-type p2p
ospf timer hello 1
ospf enable 1 area 0.0.0.0
isis enable 2
isis circuit-type p2p
isis cost 10
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.23.1.2 255.255.255.0
ospf timer hello 1
ospf enable 1 area 0.0.0.0
#
ospf 1 router-id 2.2.2.2
opaque-capability enable
area 0.0.0.0
#
route loop-detect ospf enable
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
isis 1
cost-style wide
network-entity 10.3333.0000.0001.00
#
route loop-detect isis enable
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.23.1.3 255.255.255.0
ospf timer hello 1
ospf enable 1 area 0.0.0.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.35.1.3 255.255.255.0
ospf timer hello 1
ospf enable 1 area 0.0.0.0
isis enable 1
isis circuit-type p2p
isis cost 10
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.34.1.3 255.255.255.0
isis enable 1
isis circuit-type p2p
isis cost 10
#
ospf 1 router-id 3.3.3.3
import-route isis 1 cost 100 type 2
opaque-capability enable
area 0.0.0.0
#
route loop-detect ospf enable
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
isis 1
cost-style wide
network-entity 10.4444.0000.0001.00
```

```

route loop-detect isis enable

interface LoopBack1
ip address 10.1.1.1 255.255.255.255
isis enable 1
isis cost 10

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.45.1.4 255.255.255.0
isis enable 1
isis circuit-type p2p
isis cost 10

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.34.1.4 255.255.255.0
isis enable 1
isis circuit-type p2p
isis cost 10

route loop-detect ospf enable

return
```

- DeviceE configuration file

```

sysname DeviceE

isis 1
cost-style wide
network-entity 10.5555.0000.0001.00

route loop-detect isis enable

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.56.1.5 255.255.255.0
ospf timer hello 1
ospf enable 1 area 0.0.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.35.1.5 255.255.255.0
ospf timer hello 1
ospf enable 1 area 0.0.0
isis enable 1
isis circuit-type p2p
isis cost 10

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.45.1.5 255.255.255.0
isis enable 1
isis circuit-type p2p
isis cost 10

ospf 1 router-id 5.5.5.5
import-route isis 1 cost 100 type 2
opaque-capability enable
area 0.0.0

route loop-detect ospf enable

return
```

- DeviceF configuration file

```

sysname DeviceF
```

```

isis 2
cost-style wide
network-entity 10.6666.0000.0001.00

route loop-detect isis enable

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.16.1.6 255.255.255.0
isis enable 2
isis circuit-type p2p
isis cost 10

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.26.1.6 255.255.255.0
ospf network-type p2p
ospf timer hello 1
ospf enable 1 area 0.0.0.0
isis enable 2
isis circuit-type p2p
isis cost 10

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.56.1.6 255.255.255.0
ospf timer hello 1
ospf enable 1 area 0.0.0.0

ospf 1 router-id 6.6.6.6
import-route isis 2 cost 10 type 2
opaque-capability enable
area 0.0.0.0

route loop-detect ospf enable

return
```

## Example for Configuring IS-IS Route Summarization

This section provides an example showing how to configure route summarization.

### Networking Requirements

On a medium-or large-scale IS-IS network, excessive routing entries consume a large number of system resources during route calculation and management. Therefore, route summarization needs to be configured to summarize routes with the same IPv4 prefix into one route, which effectively reduces the size of the routing table, in this way, the usage of system resources is reduced.

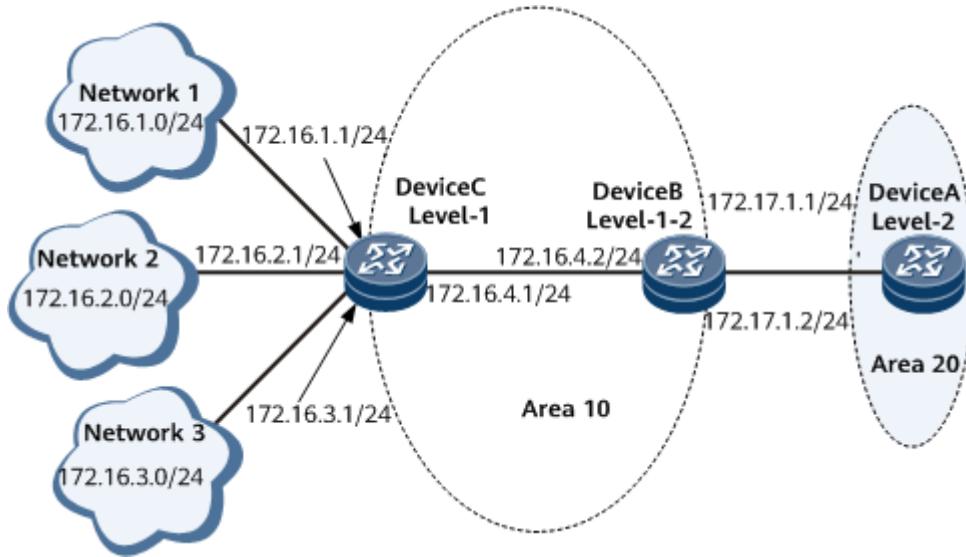
[Figure 1-291](#) shows the networking.

- Device A, Device B, and Device C run IS-IS to communicate with each other.
- Device A belongs to area 20. Device B and Device C belong to area 10.
- Device A is a Level-2 device, Device B is a Level-1-2 device, and Device C is a Level-1 device.
- Device B maintains both Level-1 and Level-2 LSDBs and leaks the routes to network segments 172.16.1.0/24, 172.16.2.0/24, and 172.16.3.0/24 from the Level-1 area to the Level-2 area. If the interface with the IP address 172.16.1.1/24 on Device C frequently alternates between up and down, the changes will be advertised to the Level-2 area, LSPs will be frequently

flooded, and DeviceB will frequently calculate routes using the SPF algorithm, causing high CPU usage on DeviceB or even network flapping.

To solve this problem, configure Device B to summarize routes to the three network segments into the route 172.16.0.0/16. This summarization reduces the number of routes to be maintained by Device B and prevents interface-state alteration in the Level-1 area from affecting route convergence in the Level-2 area.

**Figure 1-291 IS-IS route summarization networking**



| Device Name | Interface | IP Address    |
|-------------|-----------|---------------|
| Device A    | GE 2/0/0  | 172.17.1.1/24 |
| Device B    | GE 1/0/0  | 172.16.4.2/24 |
|             | GE 2/0/0  | 172.17.1.2/24 |
| Device C    | GE 1/0/0  | 172.16.4.1/24 |

## Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable IS-IS on each router and check the IS-IS routing table.
2. Configure Device B to summarize routes and check the routing table.

## Data Preparation

To complete the configuration, you need area ID of Device A (20), area ID of Device B and Device C (10), and system IDs starting from 0000.0000.0001.

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic IS-IS functions.

# Configure Device A.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-2
[*DeviceA-isis-1] network-entity 20.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] isis enable 1
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

# Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] isis enable 1
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

# Configure Device C.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] is-level level-1
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] commit
[~DeviceC-GigabitEthernet1/0/0] quit
```

The configurations of GE 2/0/0, GE 3/0/0, and GE 1/0/1 are similar to those of GE 1/0/0. For configuration details, see [Configuration Files](#) in this section.

# After completing the configurations, check the IS-IS routing table of routerDeviceA.

```
[~DeviceA] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-2 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 172.16.1.0/24 30 NULL GE2/0/0 172.17.1.2 A/-L/-
 172.16.2.0/24 30 NULL GE2/0/0 172.17.1.2 A/-L/-
 172.16.3.0/24 30 NULL GE2/0/0 172.17.1.2 A/-L/-
 172.16.4.0/24 20 NULL GE2/0/0 172.17.1.2 A/-L/-
 172.17.1.0/24 10 NULL GE2/0/0 Direct D/-L/-
Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
```

U-Up/Down Bit Set  
Protect Type: L-Link Protect, N-Node Protect

**Step 3** Configure route summarization on Device B.

```
Summarize route 172.16.1.0/24, 172.16.2.0/24, 172.16.3.0./24, and 172.16.4.0/24
into 172.16.0.0/16 on Device B.
```

```
[~DeviceB] isis 1
[~DeviceB-isis-1] summary 172.16.0.0 255.255.0.0 level-1-2
[*DeviceB-isis-1] commit
[~DeviceB-isis-1] quit
```

**Step 4** Verify the configuration.

```
Display the routing table of Device A. The following command output shows
that routes 172.16.1.0/24, 172.16.2.0/24, 172.16.3.0/24, and 172.16.4.0/24 have
been summarized into 172.16.0.0/16.
```

```
[~DeviceA] display isis route
Route information for ISIS(1)

ISIS(1) Level-2 Forwarding Table

IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

172.16.0.0/16 20 NULL GE2/0/0 172.17.1.2 A/-L/-
172.17.1.0/24 10 NULL GE2/0/0 Direct D/-L/-
Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
U-Up/Down Bit Set
Protect Type: L-Link Protect, N-Node Protect
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
is-level level-2
network-entity 20.0000.0000.0001.00

interface GigabitEthernet2/0/0
undo shutdown
ip address 172.17.1.1 255.255.255.0
isis enable 1

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
network-entity 10.0000.0000.0002.00
summary 172.16.0.0 255.255.0.0 level-1-2

interface GigabitEthernet2/0/0
undo shutdown
ip address 172.17.1.2 255.255.255.0
isis enable 1

interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.4.2 255.255.255.0
```

- ```
isis enable 1
#
return

● Device C configuration file
#
sysname DeviceC
#
isis 1
is-level level-1
network-entity 10.0000.0000.0003.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.4.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.2.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 172.16.3.1 255.255.255.0
isis enable 1
#
return
```

Example for Configuring Local MT

This section provides an example showing how to configure local MT on IS-IS networks so that multicast packets can be transmitted on a TE tunnel.

Networking Requirements

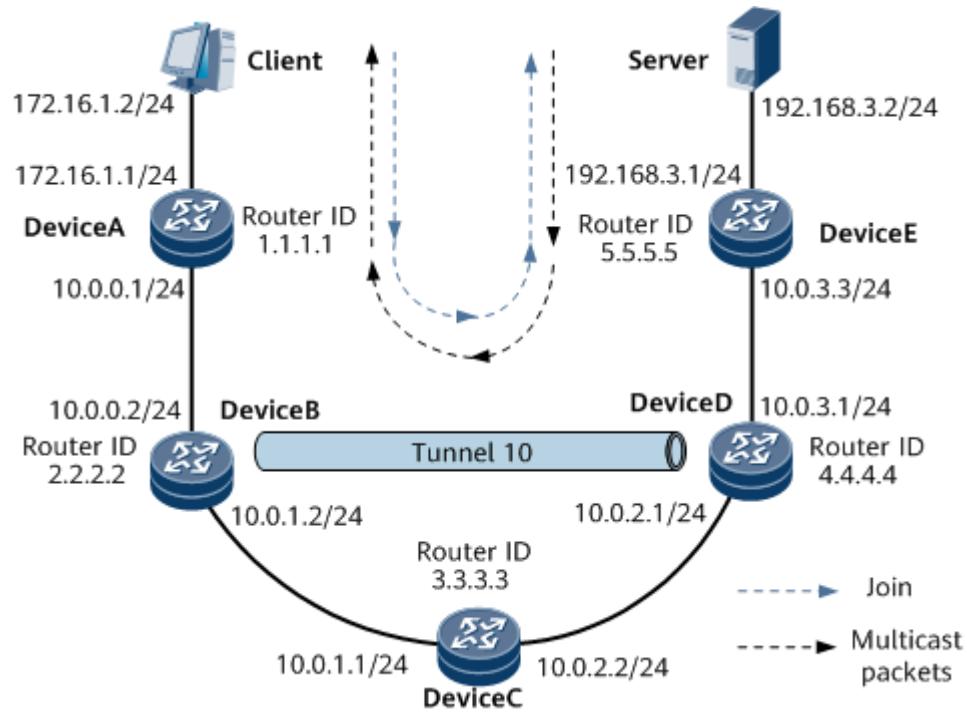
When both multicast and IGP shortcut-capable MPLS TE tunnel are configured on a network, the outbound interface of the route calculated by IS-IS using SPF may not be a physical interface but a TE tunnel interface that is up. Multicast packets are forwarded through the TE tunnel according to the IS-IS routing table. Consequently, the router bypassed by the TE tunnel is unaware of the multicast packets, unable to generate multicast forwarding entries. As a result, these multicast packets are discarded.

Local MT allows a separate multicast topology to be created on the local device. When the outbound interface of a route calculated by an IGP is an IGP Shortcut-enabled (AA) TE tunnel interface, one or a group of physical outbound interfaces are calculated for the route, resolving the conflict between a TE tunnel and multicast.

[Figure 1-292](#) shows IS-IS local MT networking.

- DeviceA, DeviceB, DeviceC, DeviceD, and DeviceE run IS-IS, and they are Level-2 devices.
- A TE tunnel is established between DeviceB and DeviceD.
- IGP Shortcut is enabled on DeviceB.

Figure 1-292 Local IS-IS MT networking



Device Name	Interface	IP Address
DeviceA	GE1/0/0	172.16.1.1/24
	GE2/0/0	10.0.0.1/24
DeviceB	GE1/0/0	10.0.0.2/24
	GE2/0/0	10.0.1.2/24
DeviceC	GE1/0/0	10.0.1.1/24
	GE2/0/0	10.0.2.2/24
DeviceD	GE1/0/0	10.0.3.1/24
	GE2/0/0	10.0.2.1/24
DeviceE	GE1/0/0	10.0.3.3/24
	GE2/0/0	192.168.3.1/24

Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

Configuration Roadmap

The configuration roadmap is as follows:

1. Enable basic IS-IS functions on each router.
2. Configure the Protocol Independent Multicast Sparse Mode (PIM-SM).
3. Configure an MPLS Resource Reservation Protocol (RSVP) TE tunnel and enable IGP Shortcut.
4. Enable local MT.

Data Preparation

To complete the configuration, you need the following data:

- IP address of each router interface (as shown in [Figure 1-292](#)), area address 10, system ID starting from 0000.0000.0001, and level (Level-2) of the routers
- Tunnel interface TE Tunnel 10, IP address of Loopback 0, tunnel encapsulation protocol MPLS TE, destination address 4.4.4.4, and tunnel ID 100

Procedure

Step 1 Assign an IP address for each interface and enable IS-IS.

In [Figure 1-292](#), assign an IP address and the mask for each interface and enable IS-IS. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure PIM-SM.

Enable multicast on all routers and enable PIM-SM on all interfaces except GigabitEthernet1/0/0 on DeviceA. The configurations on DeviceB, DeviceC, DeviceD, and DeviceE are similar to those on DeviceA. Configuration details are not mentioned here.

```
[~DeviceA] multicast routing-enable
[*DeviceA] interface gigabitethernet 2/0/0
[*DeviceA-GigabitEthernet2/0/0] pim sm
[*DeviceA-GigabitEthernet2/0/0] commit
[~DeviceA-GigabitEthernet2/0/0] quit
```

Enable IGMP on the interface through which DeviceA is connected to hosts.

```
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] igmp enable
[~DeviceA-GigabitEthernet1/0/0] igmp version 3
[*DeviceA-GigabitEthernet1/0/0] commit
```

Configure a C-BSR and a C-RP. Set the service range of the RP on DeviceD and specify the locations of the C-BSR and the C-RP.

```
[~DeviceD] pim
[*DeviceD-pim] c-bsr LoopBack0
[*DeviceD-pim] c-rp LoopBack0
[*DeviceD-pim] commit
```

Run the **display multicast routing-table** command to view the multicast routing table of a router. DeviceC is used as an example:

```
[~DeviceC] display multicast routing-table
Multicast routing table of VPN-Instance: public net
Total 1 entry
```

```
00001. (192.168.3.2, 224.31.31.31)
Uptime: 15:03:04
Upstream Interface: GigabitEthernet2/0/0
List of 1 downstream interface
 1: GigabitEthernet1/0/0
```

Step 3 Configure an MPLS RSVP-TE tunnel.

Configure DeviceB.

```
[~DeviceB] mpls lsr-id 2.2.2.2
[*DeviceB] mpls
[*DeviceB-mpls] mpls te
[*DeviceB-mpls] mpls rsvp-te
[*DeviceB-mpls] mpls te cspf
[*DeviceB-mpls] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] mpls
[*DeviceB-GigabitEthernet2/0/0] mpls te
[*DeviceB-GigabitEthernet2/0/0] mpls rsvp-te
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] isis 1
[*DeviceB-isis-1] cost-style wide
[*DeviceB-isis-1] traffic-eng level-2
[*DeviceB-isis-1] commit
[~DeviceB-isis-1] quit
```

Configure DeviceC.

```
[~DeviceC] mpls lsr-id 3.3.3.3
[*DeviceC] mpls
[*DeviceC-mpls] mpls te
[*DeviceC-mpls] mpls rsvp-te
[*DeviceC-mpls] mpls te cspf
[*DeviceC-mpls] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] mpls
[*DeviceC-GigabitEthernet1/0/0] mpls te
[*DeviceC-GigabitEthernet1/0/0] mpls rsvp-te
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] mpls
[*DeviceC-GigabitEthernet2/0/0] mpls te
[*DeviceC-GigabitEthernet2/0/0] mpls rsvp-te
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] isis 1
[*DeviceC-isis-1] cost-style wide
[*DeviceC-isis-1] traffic-eng level-2
[*DeviceC-isis-1] commit
[~DeviceC-isis-1] quit
```

Configure DeviceD.

```
[~DeviceD] mpls lsr-id 4.4.4.4
[*DeviceD] mpls
[*DeviceD-mpls] mpls te
[*DeviceD-mpls] mpls rsvp-te
[*DeviceD-mpls] mpls te cspf
[*DeviceD-mpls] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] mpls
[*DeviceD-GigabitEthernet1/0/0] mpls te
[*DeviceD-GigabitEthernet1/0/0] mpls rsvp-te
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] isis 1
[*DeviceD-isis-1] cost-style wide
[*DeviceD-isis-1] traffic-eng level-2
[*DeviceD-isis-1] commit
[~DeviceD-isis-1] quit
```

Configure an MPLS TE tunnel and enable IGP Shortcut.

Configure an MPLS TE tunnel on DeviceB and enable IGP Shortcut.

```
[~DeviceB] interface Tunnel 10
[~DeviceB-Tunnel10] ip address unnumbered interface loopback 0
[*DeviceB-Tunnel10] tunnel-protocol mpls te
[*DeviceB-Tunnel10] destination 4.4.4.4
[*DeviceB-Tunnel10] mpls te tunnel-id 100
[*DeviceB-Tunnel10] mpls te igrp shortcut isis
[*DeviceB-Tunnel10] mpls te igrp metric relative -10
[*DeviceB-Tunnel10] isis enable 1
[*DeviceB-Tunnel10] commit
[~DeviceB-Tunnel10] quit
```

Display the routing table on DeviceB. You can find that IGP Shortcut is enabled.

```
[~DeviceB] display isis route
    Route information for ISIS(1)
    -----
    ISIS(1) Level-2 Forwarding Table
    -----
    IPV4 Destination   IntCost   ExtCost ExitInterface   NextHop   Flags
    -----
    3.3.3.3/32        10        NULL     GE2/0/0      10.0.1.1   A/-/-/
    172.16.1.0/24     20        NULL     GE1/0/0      10.0.0.1   A/-/-/
    2.2.2.2/32        0         NULL     Loop0      Direct      D/-L/-/
    192.168.3.0/24    25        NULL     Tun1/0/0    2.2.2.2    A/S/-/-
    5.5.5.32          15        NULL     Tun1/0/0    2.2.2.2    A/S/-/-
    10.0.0.0/24       10        NULL     GE1/0/0      Direct      D/-L/-/
    10.0.1.0/24       10        NULL     GE2/0/0      Direct      D/-L/-/
    4.4.4.4/32        5         NULL     Tun1/0/0    2.2.2.2    A/S/-/-
    10.0.2.0/24       15        NULL     Tun1/0/0    2.2.2.2    A/S/-/-
    10.0.3.0/24       15        NULL     Tun1/0/0    2.2.2.2    A/S/-/-
Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
U-Up/Down Bit Set
Protect Type: L-Link Protect, N-Node Protect
```

Check the multicast routing table on DeviceC bypassed by the TE tunnel.

```
[~DeviceC] display multicast routing-table
```

No multicast routing entry is displayed, indicating that multicast packets have been discarded.

Step 4 Configure local MT.

Enable local MT on DeviceB.

```
[~DeviceB] isis
[~DeviceB-isis-1] local-mt enable
[*DeviceB-isis-1] commit
```

Step 5 Verify the configuration.

Display the multicast routing table on DeviceC again. The command output shows that multicast routes are displayed.

```
[~DeviceC] display multicast routing-table
Multicast routing table of VPN-Instance: public net
Total 1 entry
00001. (192.168.3.2, 224.31.31.31)
    Uptime: 00:00:19
    Upstream Interface: GigabitEthernet2/0/0
    List of 1 downstream interface
        1: GigabitEthernet1/0/0
```

Display the MIGP routing table on DeviceB.

```
[~DeviceB] display migm routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table: MIGP
Destinations : 5      Routes : 5
Destination/Mask Proto Pre Cost   Flags NextHop     Interface
  4.4.4.32/32  ISIS 15 20       10.0.1.1    GE2/0/0
  5.5.5.32/32  ISIS 15 30       10.0.1.1    GE2/0/0
  10.0.2.0/24  ISIS 15 20      10.0.1.1    GE2/0/0
  10.0.3.0/24  ISIS 15 30      10.0.1.1    GE2/0/0
  192.168.3.0/24 ISIS 15 40      10.0.1.1    GE2/0/0
```

The MIGP routing table shows that the original outbound interface (TE tunnel interface) has been replaced with a physical interface.

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA  
#  
router id 1.1.1.1  
#  
multicast routing-enable  
#  
isis 1  
is-level level-2  
cost-style wide  
network-entity 10.0000.0000.0001.00  
#  
interface GigabitEthernet2/0/0  
ip address 10.0.0.1 255.255.255.0  
isis enable 1  
pim sm  
#  
interface GigabitEthernet1/0/0  
ip address 172.16.1.1 255.255.255.0  
isis enable 1  
igmp enable  
igmp version 3  
#  
interface LoopBack0  
ip address 1.1.1.1 255.255.255.255  
#  
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
router id 2.2.2.2  
#  
multicast routing-enable  
#  
mpls lsr-id 2.2.2.2  
#  
mpls  
mpls te  
mpls rsvp-te  
mpls te cspf  
#  
isis 1  
is-level level-2  
cost-style wide  
network-entity 10.0000.0000.0002.00  
traffic-eng level-2
```

```
local-mt enable
#
interface GigabitEthernet1/0/0
ip address 10.0.0.2 255.255.255.0
isis enable 1
pim sm
#
interface GigabitEthernet2/0/0
ip address 10.0.1.2 255.255.255.0
isis enable 1
pim sm
mpls
mpls te
mpls rsvp-te
#
interface LoopBack0
ip address 2.2.2.2 255.255.255.255
isis enable 1
pim sm
#
interface Tunnel10
ip address unnumbered interface LoopBack0
tunnel-protocol mpls te
destination 4.4.4.4
mpls te tunnel-id 100
mpls te igp shortcut isis
mpls te igp metric relative -10
isis enable 1
pim sm
#
pim
C-BSR LoopBack0
C-RP LoopBack0
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
router id 3.3.3.3
#
multicast routing-enable
#
mpls lsr-id 3.3.3.3
#
mpls
mpls te
mpls rsvp-te
mpls te cspf
#
isis 1
is-level level-2
cost-style wide
network-entity 10.0000.0000.0003.00
traffic-eng level-2
#
interface GigabitEthernet1/0/0
ip address 10.0.1.1 255.255.255.0
isis enable 1
pim sm
mpls
mpls te
mpls rsvp-te
#
interface GigabitEthernet2/0/0
ip address 10.0.2.2 255.255.255.0
isis enable 1
pim sm
mpls
```

```
mpls te
mpls rsvp-te
#
interface LoopBack0
ip address 3.3.3.3 255.255.255.255
isis enable 1
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
router id 4.4.4.4
#
multicast routing-enable
#
mpls lsr-id 4.4.4.4
#
mpls
mpls te
mpls rsvp-te
mpls te cspf
#
isis 1
is-level level-2
cost-style wide
network-entity 10.0000.0000.0004.00
traffic-eng level-2
#
interface GigabitEthernet1/0/0
ip address 10.0.3.1 255.255.255.0
isis enable 1
pim sm
#
interface GigabitEthernet2/0/0
ip address 10.0.2.1 255.255.255.0
isis enable 1
pim sm
mpls
mpls te
mpls rsvp-te
#
interface LoopBack0
ip address 4.4.4.4 255.255.255.255
isis enable 1
pim sm
#
pim
C-BSR LoopBack0
C-RP LoopBack0
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
router id 5.5.5.5
#
multicast routing-enable
#
isis 1
is-level level-2
cost-style wide
network-entity 10.0000.0000.0005.00
#
interface GigabitEthernet1/0/0
ip address 10.0.3.3 255.255.255.0
isis enable 1
```

```
pim sm
#
interface GigabitEthernet2/0/0
ip address 192.168.3.1 255.255.255.0
isis enable 1
pim sm
#
interface LoopBack0
ip address 5.5.5.5 255.255.255.255
isis enable 1
pim sm
#
return
```

Example for Configuring IPv4 IS-IS Route Recursion to IPv6 Next Hops

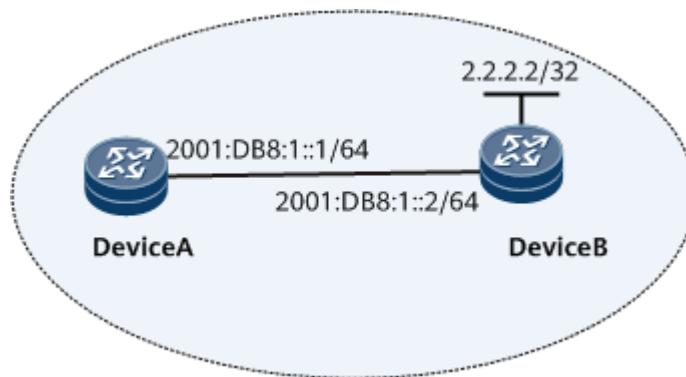
This section describes how to configure IS-IS route recursion to IPv6 next hops.

Networking Requirements

As shown in [Figure 1-293](#):

- Device A and Device B belong to the same AS, and an IS-IS neighbor relationship is established between the two devices.

Figure 1-293 Networking diagram for configuring IS-IS route recursion to IPv6 next hops



Device Name	Interface	IP Address
DeviceA	GE1/0/0	2001:DB8:1::1/64
DeviceB	Loopback0	2.2.2.2/32
	GE1/0/0	2001:DB8:1::2/64

Precautions

To improve security, you are advised to configure IS-IS authentication. For details, see "Configuring IS-IS Authentication." IS-IS interface authentication is used as an example. For details, see Example for Configuring Basic IS-IS Functions.

Configuration Roadmap

The configuration roadmap is as follows:

1. Enable IS-IS and specify NETs on Device A and Device B.
2. Configure IS-IS route recursion to IPv6 next hops on Device A, and then check the routing information.

Data Preparation

To complete the configuration, you need the following data:

- Area addresses of Device A and Device B

Procedure

Step 1 Configure basic IS-IS functions.

Configure DeviceA.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] ipv6 enable topology standard
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] ipv6 enable
[*DeviceA-GigabitEthernet1/0/0] ipv6 address 2001:DB8:1::1 64
[*DeviceA-GigabitEthernet1/0/0] isis enable 1
[*DeviceA-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

Configure DeviceB.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/0
[*DeviceB-GigabitEthernet1/0/0] ipv6 enable
[*DeviceB-GigabitEthernet1/0/0] ipv6 address 2001:DB8:1::2 64
[*DeviceB-GigabitEthernet1/0/0] isis enable 1
[*DeviceB-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

Step 2 Configure IS-IS route recursion to IPv6 next hops.

Configure DeviceA.

```
[~DeviceA] isis 1
[*DeviceA-bgp] ipv4-prefix ipv6-nexthop enable
[*DeviceA-isis-1] commit
[~DeviceA-isis-1] quit
```

Step 3 Check routing information.

Check the routing table of Device A. The command output shows that the IS-IS route recurses to an IPv6 next hop.

```
[~DeviceA] display isis route ipv4
      Route information for ISIS(1)
      -----
      ISIS(1) Level-1 Forwarding Table
```

IPV4 Destination	IntCost	ExtCost	ExitInterface	NextHop	Flags
2.2.2.2/32	10	NULL	GigabitEthernet1/0/0	FE80::3A05:28FF:FE21:300	A/-/L/- Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut, U-Up/Down Bit Set, LP-Local Prefix-Sid Protect Type: L-Link Protect, N-Node Protect

ISIS(1) Level-2 Forwarding Table					
IPV4 Destination	IntCost	ExtCost	ExitInterface	NextHop	Flags
2.2.2.2/32	10	NULL	-	-	-/-/- Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut, U-Up/Down Bit Set, LP-Local Prefix-Sid Protect Type: L-Link Protect, N-Node Protect

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA  
#  
isis 1  
network-entity 10.0000.0000.0001.00  
ipv4-prefix ipv6-nexthop enable  
#  
ipv6 enable topology standard  
#  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:DB8:1::1/64  
isis enable 1  
isis ipv6 enable 1  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
isis 1  
network-entity 10.0000.0000.0002.00  
#  
ipv6 enable topology standard  
#  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ipv6 enable  
ipv6 address 2001:DB8:1::2/64  
isis enable 1  
isis ipv6 enable 1  
#  
interface LoopBack0  
ip address 2.2.2.2 255.255.255.255  
isis enable 1  
#  
return
```

1.1.9 BGP Configuration

1.1.9.1 BGP Description

1.1.9.1.1 Overview of BGP

BGP Definition

Border Gateway Protocol (BGP) is a dynamic routing protocol used between autonomous systems (ASs).

As three earlier-released versions of BGP, BGP-1, BGP-2, and BGP-3 are used to exchange reachable inter-AS routes, establish inter-AS paths, avoid routing loops, and apply routing policies between ASs.

Currently, BGP-4 is used.

As an exterior routing protocol on the Internet, BGP has been widely used among Internet service providers (ISPs).

BGP has the following characteristics:

- Unlike an Interior Gateway Protocol (IGP), such as Open Shortest Path First (OSPF) and Routing Information Protocol (RIP), BGP is an Exterior Gateway Protocol (EGP) which controls route advertisement and selects optimal routes between ASs rather than discovering or calculating routes.
- BGP uses Transport Control Protocol (TCP) as the transport layer protocol, which enhances BGP reliability.
 - BGP selects inter-AS routes, which poses high requirements on stability. Therefore, using TCP enhances BGP's stability.
 - BGP peers must be logically connected through TCP. The destination port number is 179 and the local port number is a random value.
- BGP supports Classless Inter-Domain Routing (CIDR).
- When routes are updated, BGP transmits only the updated routes, which reduces bandwidth consumption during BGP route distribution. Therefore, BGP is applicable to the Internet where a large number of routes are transmitted.
- BGP is a distance-vector routing protocol.
- BGP is designed to prevent loops.
 - Between ASs: BGP routes carry information about the ASs along the path. The routes that carry the local AS number are discarded to prevent inter-AS loops.
 - Within an AS: BGP does not advertise routes learned in an AS to BGP peers in the AS to prevent intra-AS loops.
- BGP provides many routing policies to flexibly select and filter routes.
- BGP provides a mechanism for preventing route flapping, improving Internet stability.
- BGP can be easily extended to adapt to network development.

BGP4+ Definition

As a dynamic routing protocol used between ASs, BGP4+ is an extension of BGP.

Traditional BGP4 manages IPv4 routing information but does not support the inter-AS transmission of packets encapsulated by other network layer protocols (such as IPv6).

To support IPv6, BGP4 must have the additional ability to associate an IPv6 protocol with the next hop information and network layer reachable information (NLRI).

Two NLRI attributes that were introduced to BGP4+ are as follows:

- Multiprotocol Reachable NLRI (MP_REACH_NLRI): carries the set of reachable destinations and the next hop information.

Figure 1-294 MP_REACH_NLRI structure

AFI (2 octets)
SAFI (1 octet)
Length of Next Hop Network Address (1 octet)
Network Address of Next Hop (variable)
Number of SNPAs (1 octet)
Network Layer Reachability Information (variable)

Table 1-93 Fields in MP_REACH_NLRI

Field	Length	Description
AFI (address family identifier)	2 bytes	Indicates the address class to which the network layer protocol belongs. It is used to specify the carried IPv6 reachable route information.
SAFI (subsequent address family identifier)	1 byte	Indicates that the attribute carries information about reachable IPv6 unicast routes.
Length of Next Hop Network Address	1 byte	Indicates the length of bytes occupied by the next hop. The value 16 indicates that the link-local address is not included, and the value 32 indicates that the link-local address is included.
Network Address of Next Hop	Variable	Indicates information about the next-hop address to the destination network, which may contain a link-local address.
Number of SNPAs	1 byte	Indicates a reserved bit. The value is 0.
Network Layer Reachability Information	Variable	Indicates carried IPv6 reachable route information, including the IPv6 prefix.

- Multiprotocol Unreachable NLRI (MP_UNREACH_NLRI): carries the set of unreachable destinations.

Figure 1-295 MP_UNREACH_NLRI structure

AFI (2 octets)
SAFI (1 octet)
Network Layer UnReachability Information (1 octet)

Table 1-94 Fields in MP_UNREACH_NLRI

Field	Length	Description
AFI (Address Family Identifier)	2 bytes	Indicates that the attribute carries information about unreachable IPv6 unicast routes.
SAFI (subsequent address family identifier)	1 byte	Indicates that the attribute carries information about unreachable IPv6 unicast routes.
Network Layer UnReachability Information	1 byte	Indicates the carried IPv6 unreachable route information.

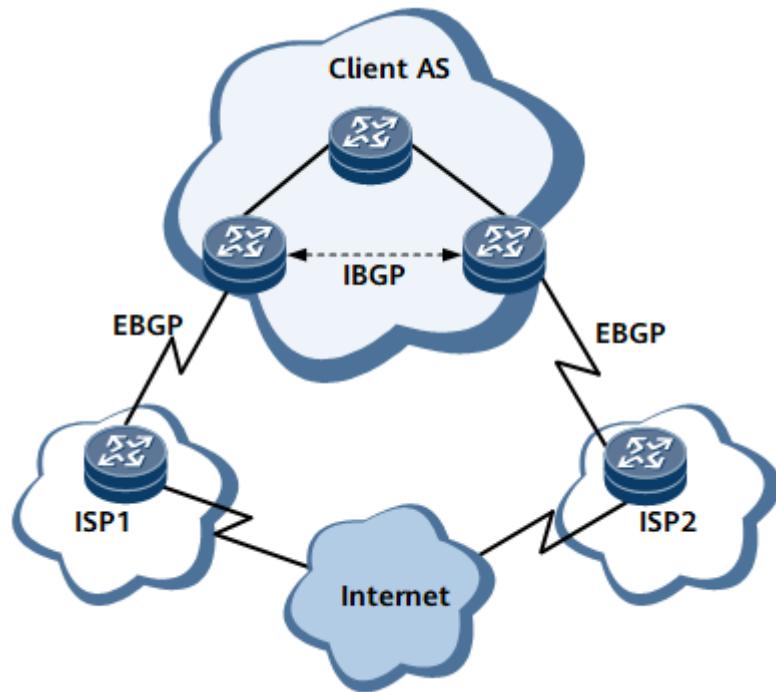
The Next_Hop attribute in BGP4+ is in the format of an IPv6 address, which can be either a globally unique IPv6 address or a next hop link-local address.

Using multiple protocol extensions of BGP, BGP4+ is applicable to IPv6 networks without changing the messaging and routing mechanisms of BGP.

Purpose

BGP transmits route information between ASs. It, however, is not required in all scenarios.

Figure 1-296 BGP networking



BGP is required in the following scenarios:

- On the network shown in [Figure 1-296](#), users need to be connected to two or more ISPs. The ISPs need to provide all or part of the Internet routes for the users. Devices, therefore, need to select the optimal route through the AS of an ISP to the destination based on the attributes carried in BGP routes.
- The AS_Path attribute needs to be transmitted between users in different organizations.
- Users need to transmit VPN routes through a Layer 3 VPN. For details, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - VPN*.
- Users need to transmit multicast routes to construct a multicast topology. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - IP Multicast*.

BGP is not required in the following scenarios:

- Users are connected to only one ISP.
- The ISP does not need to provide Internet routes for users.
- ASs are connected through default routes.

1.1.9.1.2 Understanding BGP

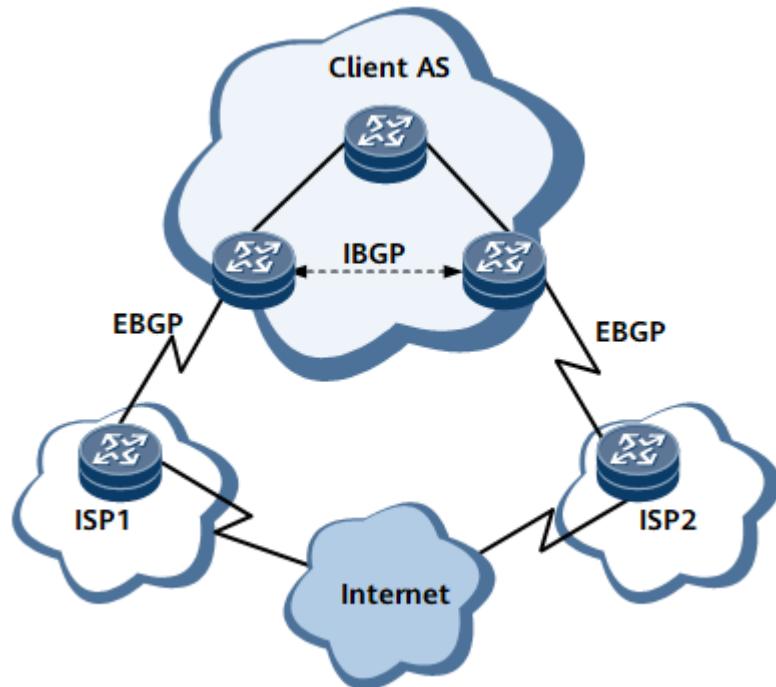
BGP Fundamentals

BGP Operating Modes

BGP runs in either of the following modes on the process the router, as shown in [Figure 1-297](#):

- Internal BGP (IBGP): When BGP runs within an AS, it is called IBGP.
- External BGP (EBGP): When BGP runs between ASs, it is called EBGP.

Figure 1-297 BGP operating modes



Roles in Transmitting BGP Messages

- Speaker: Any router that sends BGP messages is called a BGP speaker. The speaker receives or generates new routing information and then advertises the routing information to other BGP speakers. After receiving a route from another AS, the BGP speaker compares the route with its local routes. If the route is better than its local routes or is new, the speaker advertises it to all other BGP speakers except the one that advertised the route.
- Peer: BGP speakers that exchange messages with each other are called peers.

BGP Messages

BGP runs by sending five types of messages: Open, Update, Notification, Keepalive, and Route-refresh.

- Open: first message sent after a TCP connection is set up. An Open message is used to set up a BGP peer relationship. After a peer receives the Open message and negotiation between the local device and peer succeeds, the

peer sends a Keepalive message to confirm and maintain the peer relationship. After confirming that the connection is valid, peers can exchange Update, Notification, Keepalive, and Route-refresh messages.

- Update: This type of message is used to exchange routes between peers. An Update message can advertise multiple reachable routes with the same attributes and can be used to delete multiple unreachable routes.
 - An Update message can be used to advertise multiple reachable routes that share the same set of attributes. These attributes are applicable to all destinations (expressed by IP prefixes) in the network layer reachability information (NLRI) field of the Update message.
 - An Update message can be used to withdraw multiple unreachable routes. Each route is identified by its destination address (using the IP prefix), which identifies the routes previously advertised between BGP speakers.
 - An Update message can be used only to delete routes. In this case, it does not need to carry the route attributes or NLRI. When an Update message is used only to advertise reachable routes, it does not need to carry information about routes to be withdrawn.
- Notification: If BGP detects an error, it sends a Notification message to its peers. The BGP connections are then torn down immediately.
- Keepalive: BGP periodically sends Keepalive messages to peers to ensure the validity of BGP connections.
- Route-refresh: This type of message is used to request that peers re-send all reachable routes to the local device.

If all BGP devices are enabled with the route-refresh capability and an import routing policy changes, the local device sends Route-refresh messages to its peers. Upon receipt, the peers re-send their routing information to the local device. This ensures that the local BGP routing table is dynamically updated and the new routing policy is used without tearing down BGP connections.

BGP Finite State Machine

The BGP Finite State Machine (FSM) has six states: Idle, Connect, Active, OpenSent, OpenConfirm, and Established.

Three common states during the establishment of BGP peer relationships are Idle, Active, and Established.

- In the Idle state, BGP denies all connection requests. This is the initial status of BGP.
- In the Connect state, BGP decides subsequent operations after a TCP connection is established.
- In the Active state, BGP attempts to set up a TCP connection. This is the intermediate status of BGP.
- In the OpenSent state, BGP is waiting for an Open message from the peer.
- In the OpenConfirm state, BGP is waiting for a Notification or Keepalive message.
- In the Established state, BGP peers can exchange Update, Route-refresh, Keepalive, and Notification messages.

A BGP peer relationship can be established only when both BGP peers are in the Established state. Both peers send Update messages to exchange routing information.

BGP Processing

- BGP adopts TCP as its transport layer protocol. Therefore, a TCP connection must be available between the peers. Then, to establish a BGP peer relationship, the peers negotiate parameters by exchanging Open messages.
- After the peer relationship is established, BGP peers exchange BGP routing tables. BGP does not require a periodic update of its routing table. Instead, Update messages are exchanged between peers to update their routing tables incrementally if BGP routes change.
- BGP sends Keepalive messages to maintain the BGP connection between peers.
- If BGP detects an error (for example, it receives an error message), BGP sends a Notification message to report the error, and the BGP connection is torn down accordingly.

BGP Attributes

BGP route attributes are a set of parameters that describe specific BGP routes, and BGP can filter and select routes based on these attributes. BGP route attributes are classified into the following four types:

- Well-known mandatory: This type of attribute can be identified by all BGP devices and must be carried in Update messages. Without this attribute, errors occur in the routing information.
- Well-known discretionary: This type of attribute can be identified by all BGP devices. As this type of attribute is optional, it is not necessarily carried in Update messages.
- Optional transitive: This indicates the transitive attribute between ASs. A BGP device may not recognize this type of attribute, but will still accept messages carrying it and advertise them to other peers.
- Optional non-transitive: If a BGP device does not recognize this type of attribute, the device ignores it and does not advertise messages carrying it to other peers.

The most common BGP route attributes are as follows:

- Origin is a well-known mandatory attribute. It defines the origin of path information and identifies how a route becomes a BGP route. The value of the Origin attribute can be any of the following three ones:
 - IGP: This type of attribute has the highest priority. IGP is the Origin attribute for routes obtained through an IGP in the AS from which the routes originate. For example, the Origin attribute of the routes imported to the BGP routing table using the **network** command is IGP.
 - EGP: This type of attribute has the second highest priority. The Origin attribute of the routes obtained through EGP is EGP.
 - Incomplete: This type of attribute has the lowest priority and indicates the routes learned through other modes. For example, the Origin

attribute of the routes imported by BGP using the **import-route** command is Incomplete.

- AS_Path is a well-known mandatory attribute. It records the numbers of all ASs through which a route passes from the local end to the destination in the vector order.

When a BGP speaker advertises a local route:

- When advertising the route beyond the local AS, the BGP speaker adds the local AS number to the AS_Path list and then advertises this attribute to peer routers through Update messages.
- When advertising the route within the local AS, the BGP speaker creates an empty AS_Path list in an Update message.

When a BGP speaker advertises a route learned from the Update message of another BGP speaker:

- When advertising the route beyond the local AS, the BGP speaker adds the local AS number to the far left side of the AS_Path list. From the AS_Path attribute, the BGP device that receives the route learns the ASs through which the route passes to the destination. The number of the AS closest to the local AS is placed on the far left side of the list, and the other AS numbers are listed next to the former in sequence.
- When advertising the route within the local AS, the BGP speaker does not change its AS_Path attribute.

The AS_Path attribute has four types: AS_Sequence, AS_Set, AS_Confederation_Sequence, and AS_Confederation_Set.

- AS_Sequence: ordered set of ASs that the route in an Update message has traversed to reach the destination.
- AS_Set: unordered set of ASs that the route in an Update message has traversed to reach the destination. The AS_Set attribute is used in route summarization scenarios. After route summarization, the device records the unordered set of AS numbers because it cannot sequence the numbers of ASs through which specific routes pass. Regardless of how many AS numbers an AS_Set contains, BGP considers the AS_Set length to be 1 during route selection.
- AS_Confederation_Sequence: ordered set of member ASs in the local confederation. Confederation is a method of handling the surge of IBGP connections in an AS. It divides an AS into several sub-ASs. IBGP full-mesh connections are established in each sub-AS, and EBGP connections are established between sub-ASs.
- AS_Confederation_Set: unordered set of member ASs in the local confederation. The AS_Confederation_Set attribute is used in route summarization scenarios in a confederation.

The AS_Confederation_Sequence and AS_Confederation_Set attributes are used to prevent routing loops and select routes among the member ASs in a confederation.

- The Next_Hop attribute is a well-known mandatory attribute. Different from the Next_Hop attribute in an IGP, the Next_Hop attribute in BGP may not be the IP address of a neighboring router. In most cases, the Next_Hop attribute complies with the following rules:
 - When advertising a route to an EBGP peer, a BGP speaker sets the Next_Hop of the route to the address of the local interface used to establish the EBGP peer relationship.

- When advertising a locally generated route to an IBGP peer, a BGP speaker sets the Next_Hop of the route to the address of the local interface used to establish the IBGP peer relationship.
- When advertising a route learned from an EBGP peer to an IBGP peer, a BGP speaker does not change the Next_Hop of the route.
- Multi-Exit-Discriminator (MED) is an optional non-transitive attribute. It is transmitted only between two neighboring ASs. The AS that receives the MED attribute will not advertise it to any third AS.
Similar to the metric used by an IGP, the MED is used to determine the optimal route when traffic enters an AS. If the router running BGP obtains multiple routes from different EBGP peers and these routes have the same destination but different next hops, the device selects the route with the smallest MED value.
- Local_Pref is a well-known discretionary attribute that indicates the preference of a BGP route on a router. It is valid only between IBGP peers and is not advertised to other ASs.
The Local_Pref attribute determines the optimal route for the traffic that leaves an AS. When a BGP router obtains multiple routes to the same destination address but with different next hops through IBGP peers, the route with the largest Local_Pref value is selected.

BGP Message Format

A BGP message consists of a BGP header and a data portion. BGP runs by sending five types of messages: Open, Update, Notification, Keepalive, and Route-refresh. These messages use the same header format. BGP messages are transmitted based on TCP (port 179). The message length varies from 19 octets to 4096 octets. The header of each BGP message is 19 octets, consisting of three fields.

- [Open Message](#)
- [Update Message](#)
- [Notification Message](#)
- [Keepalive Message](#)
- [Route-refresh Message](#)

Message Header Format

The five types of BGP messages have the same header format. [Figure 1-298](#) shows the format of a BGP message header.

Figure 1-298 Message header format

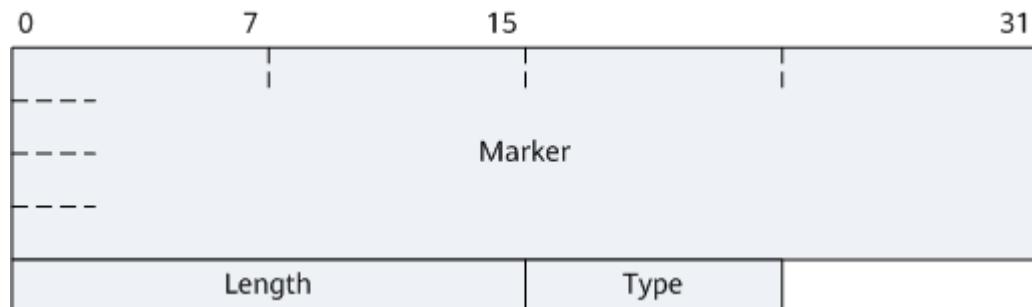


Table 1-95 Description of the header fields

Field	Length	Description
Marker	16 octets	Indicates whether the information synchronized between BGP peers is complete. This field is used for calculation in BGP authentication. If no authentication is used, the field is set to all ones in binary format or all Fs in hexadecimal notation.
Length	2 octets (unsigned integer)	Indicates the total length of a BGP message (including the header), in octets. The length ranges from 19 octets to 4096 octets.
Type	1 octet (unsigned integer)	Indicates the BGP message type, which has five values. <ul style="list-style-type: none"> ● OPEN ● UPDATE ● NOTIFICATION ● KEEPALIVE ● REFRESH

Open Message

Open messages are used to establish BGP connections. The value of the Type field in the header of an Open message is 1. [Figure 1-299](#) shows the format of an Open message.

Figure 1-299 Format of an Open message without the header

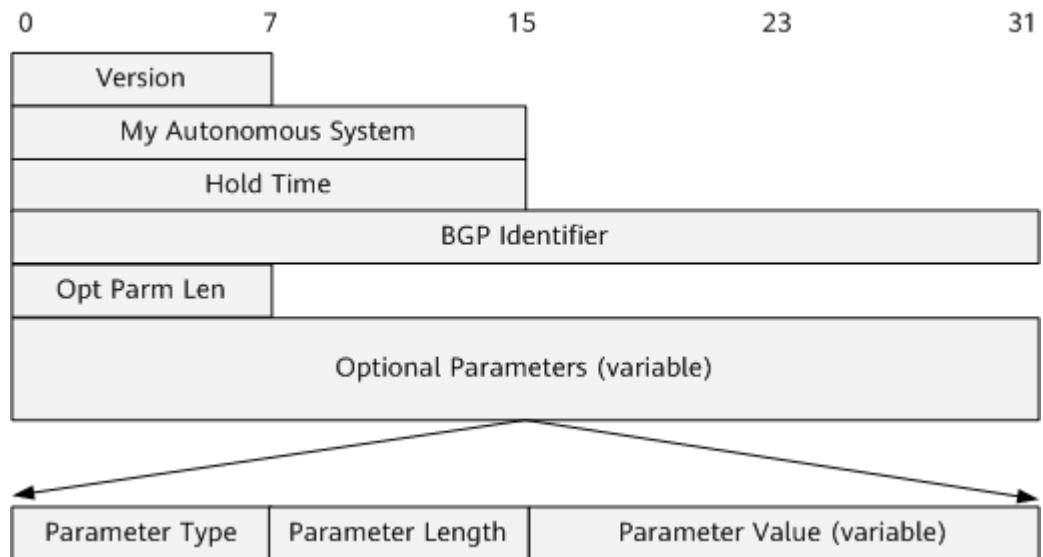


Table 1-96 Description of each field in the Open message without the header

Field	Length	Description
Version	1 octet (unsigned integer)	Indicates the BGP version number. For BGP-4, the value of the field is 4.
My Autonomous System	2 octets (unsigned integer)	Indicates the AS number of the message sender.
Hold Time	2 octets (unsigned integer)	Indicates the hold time set by the message sender, in seconds. BGP peers use this field to negotiate the interval at which Keepalive or Update messages are sent so that the peers can maintain the connection between them. Upon receipt of an Open message, the finite state machine (FSM) of a BGP speaker must compare the locally configured hold time with that carried in the received Open message. The FSM uses the smaller value as the negotiation result. The value of Hold Time can be 0 (no Keepalive message is sent) or greater than or equal to 3. The default value is 180.
BGP Identifier	4 octets (unsigned integer)	Router ID of the sender.
Opt Parm Len	1 octet (unsigned integer)	Indicates the length of the Optional Parameters field. If the value is 0, no optional parameters are used.

Field	Length	Description																																																						
Optional Parameters	Variable	<p>Indicates a list of optional BGP parameters, with each one representing a unit in TLV format.</p> <table style="margin-left: 20px;"> <tr><td>0</td><td>7</td><td>15</td><td></td></tr> <tr><td>+-----+-----+</td><td>+-----+-----+</td><td>+-----+-----+</td><td>...</td></tr> <tr><td> Parm.Type </td><td>Parm.Length </td><td>Parm.Value (variable)</td><td></td></tr> <tr><td>+-----+-----+</td><td>+-----+-----+</td><td>+-----+-----+</td><td>...</td></tr> </table> <ul style="list-style-type: none"> ● Parm.Type: indicates the parameter type. The value is an unsigned integer and occupies 1 octet. The field is valid only if its value is 2, which indicates that a capability needs to be negotiated. ● Parm.Length: indicates the length of Parameter Value. The value is an unsigned integer and occupies 1 octet. ● Parm.Value: varies with Parm.Type. If the value of Parm.Type is 2, Parm.Value indicates the list of capabilities that can be negotiated. Each unit in the list is a TLV triplet. <table style="margin-left: 20px;"> <tr><td>+-----+</td><td></td></tr> <tr><td> Capability Code (1 octet) </td><td></td></tr> <tr><td>+-----+</td><td></td></tr> <tr><td> Capability Length (1 octet) </td><td></td></tr> <tr><td>+-----+</td><td></td></tr> <tr><td> Capability Value (variable) </td><td></td></tr> <tr><td>+-----+</td><td></td></tr> </table> <ul style="list-style-type: none"> - Capability Code: indicates a capability number and occupies 1 octet. If the value is 1, the address family capability is supported. If the value is 2, the route-refresh capability is supported. - Capability Length: indicates the length of Capability Value and occupies 1 octet. - Capability Value: varies with Capability Code. If the value of Capability Code is 1: Capability Value is a TLV triplet and occupies 4 octets. <table style="margin-left: 20px;"> <tr><td>0</td><td>7</td><td>15</td><td>23</td><td>31</td><td></td></tr> <tr><td>+-----+-----+-----+-----+</td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td> AFI Res. SAFI </td><td></td><td></td><td></td><td></td><td></td></tr> <tr><td>+-----+-----+-----+</td><td></td><td></td><td></td><td></td><td></td></tr> </table> <p>AFI: short for address family identifier and 2 octets in length. AFI is used with the subsequent AFI (SAFI) to determine the relationship between the network layer protocol and IP address. The encoding mode is the same as that in multiprotocol extensions. The value complies with the address family numbers defined in the related RFC protocol.</p> <p>Res: is reserved and 1 octet in length. It must be set to 0 by the sender and is ignored when it is received.</p>	0	7	15		+-----+-----+	+-----+-----+	+-----+-----+	...	Parm.Type	Parm.Length	Parm.Value (variable)		+-----+-----+	+-----+-----+	+-----+-----+	...	+-----+		Capability Code (1 octet)		+-----+		Capability Length (1 octet)		+-----+		Capability Value (variable)		+-----+		0	7	15	23	31		+-----+-----+-----+-----+						AFI Res. SAFI						+-----+-----+-----+					
0	7	15																																																						
+-----+-----+	+-----+-----+	+-----+-----+	...																																																					
Parm.Type	Parm.Length	Parm.Value (variable)																																																						
+-----+-----+	+-----+-----+	+-----+-----+	...																																																					
+-----+																																																								
Capability Code (1 octet)																																																								
+-----+																																																								
Capability Length (1 octet)																																																								
+-----+																																																								
Capability Value (variable)																																																								
+-----+																																																								
0	7	15	23	31																																																				
+-----+-----+-----+-----+																																																								
AFI Res. SAFI																																																								
+-----+-----+-----+																																																								

Field	Length	Description
		<p>SAFI: occupies 1 octet. SAFI is used with AFI to determine the relationship between the network layer protocol and IP address. The encoding mode is the same as that in multiprotocol extensions. The value complies with the address family numbers defined in the related RFC protocol.</p> <p>If the value of Capability Code is 2:</p> <p>The route-refresh capability is supported. The code of this capability is 2, the length is 0, and there is no value.</p> <p>Devices can process Route-refresh messages only after the route-refresh capability is negotiated successfully. By default, the IPv4 unicast and route-refresh capabilities are supported.</p>

Open Message Extensions

With the enhancement of BGP capabilities, when a BGP session negotiates multiple capabilities, the length of an Open message may exceed 255 bytes. You can run the **peer extended-open-message** command to use the extended format of an Open message, which is shown in [Figure 1-300](#).

Figure 1-300 Extended format of an Open message

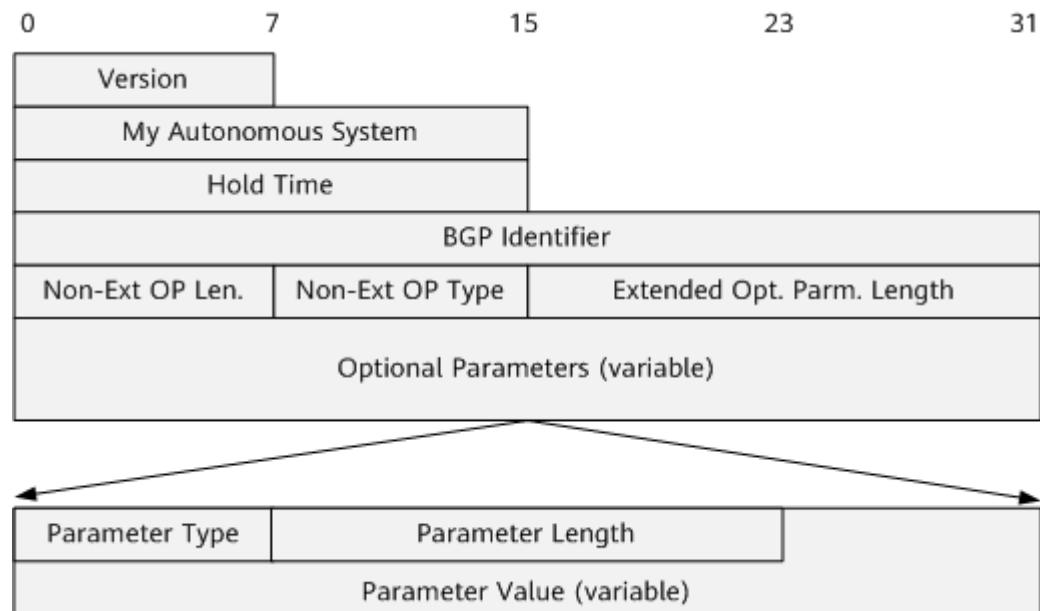


Table 1-97 Description of each field in the extended format of an Open message

Field	Length	Description
Version	1 octet (unsigned integer)	Indicates the BGP version number. For BGP-4, the value of the field is 4.
My Autonomous System	2 octets (unsigned integer)	Indicates the AS number of the message sender.
Hold Time	2 octets (unsigned integer)	Indicates the hold time set by the message sender, in seconds. BGP peers use this field to negotiate the interval at which Keepalive or Update messages are sent to maintain the connection between them. Upon receipt of an Open message, the finite state machine (FSM) of a BGP speaker must compare the locally configured hold time with that carried in the received Open message. The FSM uses the smaller value as the negotiation result. The value of Hold Time can be 0 (no Keepalive message is sent) or greater than or equal to 3. The default value is 180.
BGP Identifier	4 octets (unsigned integer)	Router ID of the sender.
Non-Ext OP Len.	1 octet (unsigned integer)	The value of this field is fixed at 255.
Non-Ext OP Type	1 octet (unsigned integer)	IANA has registered the optional parameter extension length type code 255 as an extended optional parameter type for BGP Open messages.
Extended Opt.Parm.Length	2 octets (unsigned integer)	Length of extended optional parameters.

Field	Length	Description																																																										
Optional Parameters	Variable	<p>Indicates a list of optional BGP parameters, with each one representing a unit in TLV format.</p> <table border="0"> <tr> <td>0</td> <td>7</td> <td>15</td> <td>23</td> </tr> <tr> <td>+-----+-----+-----+-----+</td> <td> Parm.Type </td> <td> </td> <td> </td> </tr> <tr> <td>+-----+-----+-----+-----+</td> <td> </td> <td>Parm.Length</td> <td> </td> </tr> <tr> <td>+-----+-----+-----+-----+</td> <td> </td> <td> </td> <td> </td> </tr> <tr> <td>~</td> <td>Parm.Value (variable)</td> <td>~</td> <td> </td> </tr> <tr> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </table> <ul style="list-style-type: none"> • Parm.Type: indicates the parameter type. The value is an unsigned integer and occupies 1 octet. The field is valid only if its value is 2, which indicates that a capability needs to be negotiated. • Parm.Length: indicates the length of Parameter Value. The value is an unsigned integer and occupies 2 octets. • Parm.Value: varies with Parm.Type. If the value of Parm.Type is 2, Parm.Value indicates the list of capabilities that can be negotiated. Each unit in the list is a TLV triplet. <table border="0"> <tr> <td>+-----+</td> <td> </td> </tr> <tr> <td> Capability Code (1 octet) </td> <td> </td> </tr> <tr> <td>+-----+</td> <td> </td> </tr> <tr> <td> Capability Length (2 octet) </td> <td> </td> </tr> <tr> <td>+-----+</td> <td> </td> </tr> <tr> <td> Capability Value (variable) </td> <td> </td> </tr> <tr> <td>+-----+</td> <td> </td> </tr> </table> <ul style="list-style-type: none"> - Capability Code: indicates a capability number and occupies 1 octet. If the value is 1, the address family capability is supported. If the value is 2, the route-refresh capability is supported. - Capability Length: indicates the length of Capability Value and occupies 2 octets. - Capability Value: varies with Capability Code. If the value of Capability Code is 1: Capability Value is a TLV triplet and occupies 4 octets. <table border="0"> <tr> <td>0</td> <td>7</td> <td>15</td> <td>23</td> <td>31</td> </tr> <tr> <td>+-----+-----+-----+-----+</td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> <tr> <td> AFI Res. SAFI </td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> <tr> <td>+-----+-----+-----+-----+</td> <td> </td> <td> </td> <td> </td> <td> </td> </tr> </table> <p>AFI: is short for address family identifier and occupies 2 octets. AFI is used with the subsequent AFI (SAFI) to determine the relationship between the network layer protocol and IP address. The encoding mode is the same as that in multiprotocol extensions. The value complies with the address family numbers defined in the related RFC protocol.</p>	0	7	15	23	+-----+-----+-----+-----+	Parm.Type			+-----+-----+-----+-----+		Parm.Length		+-----+-----+-----+-----+				~	Parm.Value (variable)	~						+-----+		Capability Code (1 octet)		+-----+		Capability Length (2 octet)		+-----+		Capability Value (variable)		+-----+		0	7	15	23	31	+-----+-----+-----+-----+					AFI Res. SAFI					+-----+-----+-----+-----+				
0	7	15	23																																																									
+-----+-----+-----+-----+	Parm.Type																																																											
+-----+-----+-----+-----+		Parm.Length																																																										
+-----+-----+-----+-----+																																																												
~	Parm.Value (variable)	~																																																										
+-----+																																																												
Capability Code (1 octet)																																																												
+-----+																																																												
Capability Length (2 octet)																																																												
+-----+																																																												
Capability Value (variable)																																																												
+-----+																																																												
0	7	15	23	31																																																								
+-----+-----+-----+-----+																																																												
AFI Res. SAFI																																																												
+-----+-----+-----+-----+																																																												

Field	Length	Description
		<p>Res: is reserved and 1 octet in length. It must be set to 0 by the sender and is ignored when it is received.</p> <p>SAFI: occupies 1 octet and is used with the AFI to determine the relationship between the network layer protocol and the IP address.</p> <p>The value 2 of Capability Code indicates that the route refresh capability is supported. The code of this capability is 2, the length is 0, and there is no value.</p> <p>Devices can process Route-refresh messages only after the route-refresh capability is negotiated successfully. By default, the device supports IPv4 unicast and route-refresh capabilities.</p>

Update Message

Update messages are used to transfer routing information between BGP peers. The value of the Type field in the header of an Update message is 2. [Figure 1-301](#) shows the format of an Update message without the header.

Figure 1-301 Format of an Update message without the header

Withdrawn Routes Length (2 octets)
Withdrawn Routes (variable)
Total Path Attribute Length (2 octets)
Path Attributes (variable)
Network Layer Reachability Information (variable)

Table 1-98 Description of each field in the Update message without the header

Field	Length	Description
Withdrawn Routes Length	2 octets (unsigned integer)	Indicates the length of the Withdrawn Routes field. If the value is 0, no route is withdrawn.

Field	Length	Description															
Withdrawn Routes	Variable	<p>Contains a list of routes to be withdrawn. Each entry in the list contains the Length (1 octet) and Prefix (length-variable) fields.</p> <ul style="list-style-type: none">Length: indicates the mask length of the route to be withdrawn. The value 0 indicates that all routes are matched.Prefix: The prefix of the transmitted IP address must be represented by an integer byte. For example, consider the withdrawal of the route 192.168.200.200. The Prefix (in hexadecimal encoding) of the route varies according to different mask lengths: <table><thead><tr><th>Mask (decimal)</th><th>Length</th><th>Prefix</th></tr></thead><tbody><tr><td>32</td><td>20</td><td>C0 A8 C8 C8</td></tr><tr><td>25</td><td>19</td><td>C0 A8 C8 80</td></tr><tr><td>20</td><td>14</td><td>C0 A8 C0</td></tr><tr><td>15</td><td>0F</td><td>C0 A8</td></tr></tbody></table>	Mask (decimal)	Length	Prefix	32	20	C0 A8 C8 C8	25	19	C0 A8 C8 80	20	14	C0 A8 C0	15	0F	C0 A8
Mask (decimal)	Length	Prefix															
32	20	C0 A8 C8 C8															
25	19	C0 A8 C8 80															
20	14	C0 A8 C0															
15	0F	C0 A8															
Total Path Attribute Length	2 octets (unsigned integer)	Indicates the total length of the Path Attributes field. If the value is 0, no route or route attributes need to be advertised.															

Field	Length	Description																									
Path Attributes	Variable	<p>Indicates a list of path attributes in the Update message. The type codes of the path attributes are arranged in ascending order. Each path attribute is encoded as a TLV (<attribute type, attribute length, attribute value>) of variable length.</p> <p>Figure 1-302 Format of the BGP path attribute TLV</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">15</td> </tr> <tr> <td style="text-align: center;">Attr. TYPE</td> <td style="text-align: center;">Attr. Length (variable)</td> <td style="text-align: center;">Attr. Value (variable)</td> </tr> </table> <p>Attr.TYPE occupies two octets (unsigned integer), including the one-octet Flags field (unsigned integer) and the one-octet Type Code field (unsigned integer).</p> <p>Figure 1-303 TLV structure-Type</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">7</td> <td style="text-align: center;">15</td> </tr> <tr> <td colspan="2"></td> <td style="text-align: center;">Attr. Type Code</td> </tr> <tr> <td colspan="3" style="text-align: center;">Attr. Flags</td> </tr> <tr> <td colspan="3" style="text-align: center;"> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 25%;">O</td> <td style="width: 25%;">T</td> <td style="width: 25%;">P</td> <td style="width: 25%;">E</td> </tr> <tr> <td>U</td> <td>U</td> <td>U</td> <td>U</td> </tr> </table> </td> </tr> </table> <p>Attr.Flags: occupies one octet (eight bits) and indicates the attribute flag. The meaning of each bit is as follows:</p> <ul style="list-style-type: none"> O (Optional bit): defines whether the attribute is optional. The value 1 indicates an optional attribute, whereas the value 0 indicates a well-known attribute. T (Transitive bit): Defines whether the attribute is transitive. For an optional attribute, the value 1 indicates that the attribute is transitive, whereas the value 0 indicates that the attribute is non-transitive. For a well-known attribute, the value must be set to 1. P (Partial bit): defines whether the attribute is partial. If the optional transitive attribute is partial, the value is set to 1; if the attribute is complete, the value is set to 0. For well-known attributes and for optional non-transitive attributes, the value must be set to 0. E (Extended Length bit): defines whether the length (Attr. Length) of the attribute needs to be extended. If the attribute length does not need to be extended, the value is set to 0 and the Attr. Length is 1 octet. If the attribute length needs to be extended, the value is set to 1 and the Attr. Length is 2 octets. 	0	15	Attr. TYPE	Attr. Length (variable)	Attr. Value (variable)	0	7	15			Attr. Type Code	Attr. Flags			<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 25%;">O</td> <td style="width: 25%;">T</td> <td style="width: 25%;">P</td> <td style="width: 25%;">E</td> </tr> <tr> <td>U</td> <td>U</td> <td>U</td> <td>U</td> </tr> </table>			O	T	P	E	U	U	U	U
0	15																										
Attr. TYPE	Attr. Length (variable)	Attr. Value (variable)																									
0	7	15																									
		Attr. Type Code																									
Attr. Flags																											
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 25%;">O</td> <td style="width: 25%;">T</td> <td style="width: 25%;">P</td> <td style="width: 25%;">E</td> </tr> <tr> <td>U</td> <td>U</td> <td>U</td> <td>U</td> </tr> </table>			O	T	P	E	U	U	U	U																	
O	T	P	E																								
U	U	U	U																								

Field	Length	Description
		<p>U (Unused bits): Indicates that the lower-order 4 bits are not used. These bits must be set to 0s upon transmission and ignored upon receipt.</p> <p>Attr.Type Code: Indicates the attribute type code and occupies 1 octet (unsigned integer). For details about the type codes, see Table 1-99.</p> <p>Attr.Value: Enter the attribute value based on the attribute type.</p>
Network Layer Reachability Information (NLRI)	Variable	Indicates a list of IP address prefixes in the Update message. Each address prefix in the list is encoded as a 2-tuple LV (<prefix length, the prefix of the reachable route>). The encoding mode is the same as that used for Withdrawn Routes.

Table 1-99 Type codes of route attributes

Attribute Type Code	Attribute Value
1: Origin	IGP, EGP, or Incomplete
2: AS_Path	AS_Set, AS_Sequence, AS_Confederation_Set, or AS_Confederation_Sequence
3: Next_Hop	Next-hop IP address.
4: Multi_Exit_Disc	MED that is used to identify the optimal route for the traffic to enter an AS.
5: Local_Pref	Local_Pref that is used to identify the optimal route for the traffic to leave an AS.
6: Atomic_Aggregate	The BGP speaker selects the summary route rather than a specific route.
7: Aggregator	Router ID and AS number of the device that performs route summarization.
8: Community	Community attribute.
9: Originator_ID	Router ID of the originator of the reflected route.
10: Cluster_List	List of the RRs through which the reflected route passes.
14: MP_REACH_NLRI	Multiprotocol reachable NLRI.
15: MP_UNREACH_NLRI	Multiprotocol unreachable NLRI.

Attribute Type Code	Attribute Value
16: Extended Communities	Extended community attribute.

Notification Message

Notification messages are used to notify BGP peers of errors in a BGP process. The value of the Type field in the header of a Notification message is 3. [Figure 1-304](#) shows the format of a Notification message without the header.

Figure 1-304 Format of a Notification message without the header

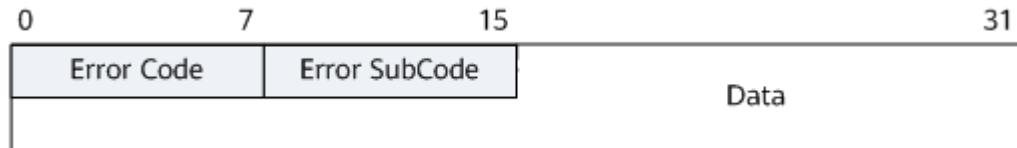


Table 1-100 Description of each field in the Notification message without the header

Field	Length	Description
Error code	1 octet	Indicates an error type. The value 0 indicates a non-specific error type. For details about the error codes, see Table 1-101 .
Error subcode	1 octet	Specifies the number of an error detail. The number of a non-specific error detail is 0.
Data	Variable	Indicates the error data.

Table 1-101 Description of the BGP error codes

Error Code	Error Subcode
1: message header error	1: Connections are not synchronized.
	2: Incorrect message length.
	3: Incorrect message type.
2: Open message error	1: Unsupported version number.
	2: Incorrect peer AS.
	3: Incorrect BGP identifier.
	4: Unsupported optional parameter.

Error Code	Error Subcode
	5: Authentication failure. 6: Unacceptable hold time. 7: Unsupported capability.
3: Update message error	1: Malformed attribute list. 2: Unrecognized well-known attribute. 3: Missing well-known attribute. 4: Incorrect attribute flag. 5: Incorrect attribute length. 6: Invalid Origin attribute. 7: AS routing loop. 8: Invalid Next_Hop attribute. 9: Incorrect optional attribute. 10: Invalid network field. 11: Malformed AS_Path.
4: The hold timer expires.	0: No special error subcode is defined.
5: FSM error	0: No special error subcode is defined. 1: An unexpected message is received in the OpenSent state. 2: An unexpected message is received in the OpenConfirm state. 3: An unexpected message is received in the Established state.
6: Cease	1: The number of prefixes exceeded the maximum. 2: Administrative shutdown. 3: The peer is deleted. 4: Administrative reset. 5: The connection fails. 6: Other configurations change. 7: Connection conflict. 8: Resource shortage. 9: The BFD session is interrupted.

Keepalive Message

Keepalive messages are used to maintain BGP connections. The value of the Type field in the header of a Keepalive message is 4. Each Keepalive message has only a BGP header; it does not have a data portion. Therefore, the total length of each Keepalive message is fixed at 19 octets.

Route-refresh Message

Route-refresh messages are used to dynamically request a BGP route advertiser to re-send Update messages. The value of the Type field in the header of a Route-refresh message is 5. [Figure 1-305](#) shows the format of a Route-refresh message without the header.

Figure 1-305 Format of a Route-refresh message without the header

0	7	15	23	31
	AFI	Res.	SAFI	

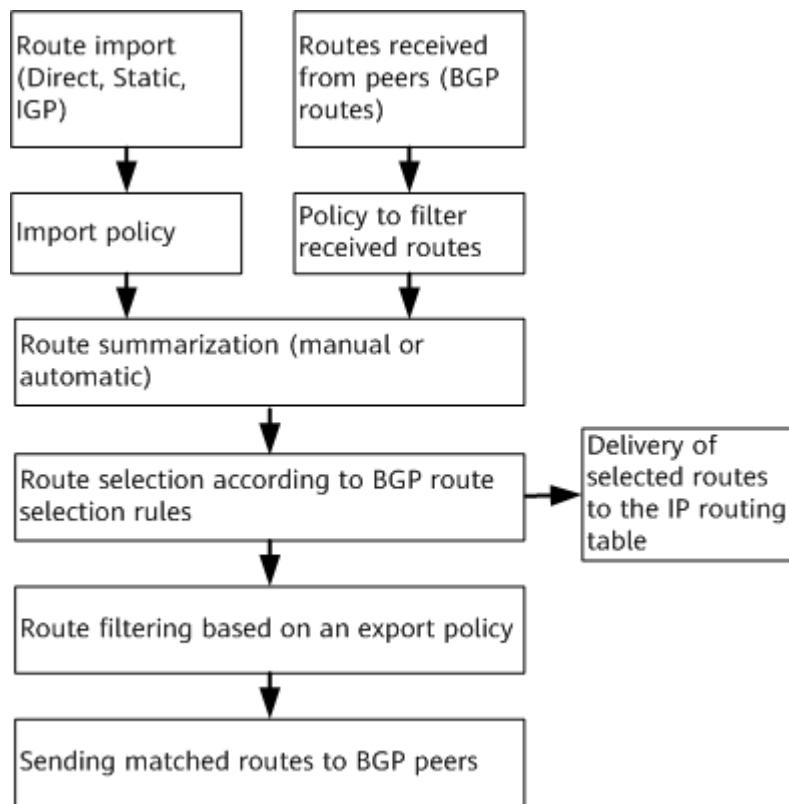
Table 1-102 Description of each field in the Route-refresh message without the header

Field	Length	Description
AFI	2 octets (unsigned integer)	Indicates the address family ID, which is defined the same as that in Open messages.
Res.	1 octet (unsigned integer)	Must be all 0s. The field is ignored upon receipt.
SAFI	1 octet (unsigned integer)	Is defined the same as that in Open messages.

BGP Route Processing

[Figure 1-306](#) shows how BGP processes routes. BGP routes can be imported from other protocols or learned from peers. To reduce the routing size, you can configure route summarization for optimal routes. In addition, you can configure route-policies and apply them to route import, receipt, or advertisement to filter routes or modify route attributes.

Figure 1-306 BGP route processing



For details about route import, see [Route Import](#); for details about BGP route selection rules, see [BGP Route Selection](#); for details about route summarization, see [Route Summarization](#); for details about rules for advertising routes to BGP peers, see [BGP Route Advertisement](#).

For details about import or export policies, see "Routing Policies" in *NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X Feature Description — IP Routing*.

For details about BGP load balancing, see Load Balancing Among BGP Routes.

Route Import

BGP itself cannot discover routes. Therefore, it needs to import other protocol routes, such as IGP routes or static routes, to the BGP routing table. Imported routes can be transmitted within an AS or between ASs.

BGP can import routes in either Import or Network mode.

- The Import mode enables BGP to import routes by protocol type, such as RIP, OSPF, IS-IS, static routes, and direct routes.
- The Network mode imports a route with the specified prefix and mask to the BGP routing table, which is more precise than the Import mode.

BGP Route Selection

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, when multiple routes to the same destination are available, BGP selects routes based on the following rules:

1. Prefers the routes that do not recurse to an SRv6 TE Policy in the Graceful Down state (the SRv6 TE Policy is in the delayed deletion state).
2. Prefers routes in descending order of Valid, Not Found, and Invalid after BGP origin AS validation results are applied to route selection in a scenario where the device is connected to a Resource Public Key Infrastructure (RPKI) server.
3. Prefers routes without bit errors.
If the **bestroute bit-error-detection** command is run, BGP preferentially selects routes without bit error events.
4. Prefers the peer routes with an IPv4 or IPv6 next hop address or remotely leaked routes.
If the **bestroute nexthop-priority ipv4 high-level** command is run, BGP prefers the routes with an IPv4 next hop address.
If the **bestroute nexthop-priority ipv6 high-level** command is run, BGP prefers the routes with an IPv6 next hop address.
5. Prefers the prefix routes that are learned from VPNV4/VPNV6 or EVPN and then are leaked to a VPN instance.
If the **bestroute address-family-priority vpng4 high-level** command is run, BGP compares the VPN address family types of routes when selecting the optimal route, and the IPv4 prefix routes leaked from VPNV4 take precedence over the IPv4 prefix routes leaked from EVPNv4.
If the **bestroute address-family-priority evpnv4 high-level** command is run, BGP compares the VPN address family types of routes when selecting the optimal route, and the IPv4 prefix routes leaked from EVPNv4 take precedence over the IPv4 prefix routes leaked from VPNV4.
If the **bestroute address-family-priority vpng6 high-level** command is run, BGP compares the VPN address family types of routes when selecting the optimal route, and the IPv6 prefix routes leaked from VPNV6 take precedence over the IPv6 prefix routes leaked from EVPNv6.
If the **bestroute address-family-priority evpnv6 high-level** command is run, BGP compares the VPN address family types of routes when selecting the optimal route, and the IPv6 prefix routes leaked from EVPNv6 take precedence over the IPv6 prefix routes leaked from VPNV6.
6. Prefers the route with the largest PrefVal value.
PrefVal is Huawei-specific. It is valid only on the device where it is configured.
7. Prefers the routes with next hops recursing to SRv6 tunnels, and prefers the routes with next hops recursing to SRv6 TE Policies over the routes with next hops recursing to SRv6 BE tunnels.
If the **bestroute nexthop-recursive-priority srv6-te-policy** command is run, the routes with next hops recursing to SRv6 TE Policies are preferentially selected.
8. Prefers the route with the largest Local_Pref value.
If a route does not carry Local_Pref, the default value 100 takes effect during BGP route selection. To change the value, run the **default local-preference** command.
9. Prefers a locally originated route to a route learned from a peer.
Locally originated routes include routes imported using the **network** or **import-route** command, as well as manually and automatically generated summary routes.

- a. Prefers a summary route over a non-summary route.
 - b. Prefers a route obtained using the **aggregate** command over a route obtained using the **summary automatic** command.
 - c. Prefers a route imported using the **network** command over a route imported using the **import-route** command.
10. Prefers a route that carries the Accumulated Interior Gateway Protocol Metric (AIGP) attribute.
- The priority of a route that carries the AIGP attribute is higher than the priority of a route that does not carry the AIGP attribute.
 - If two routes both carry the AIGP attribute, the route with a smaller AIGP attribute value plus IGP metric of the recursive next hop is preferred over the other route.
11. Prefers the route with the shortest AS_Path.
- The AS_CONFED_SEQUENCE and AS_CONFED_SET are not included in the AS_Path length.
 - During route selection, a device assumes that an AS_SET carries only one AS number regardless of the actual number of ASs it carries.
 - If the **bestroute as-path-ignore** command is run, BGP no longer compares the AS_Path attribute.

 **NOTE**

After the **load-balancing as-path-ignore** command is run, the routes with different AS_Path values can load-balance traffic.

12. Prefers the route with the Origin type as **IGP**, **EGP**, and **Incomplete** in descending order.
13. Prefers the route with the smallest MED value.

 **NOTE**

If the **bestroute med-plus-igp** command is run, BGP preferentially selects the route with the smallest sum of MED multiplied by a MED multiplier and IGP cost multiplied by an IGP cost multiplier.

- BGP compares the MEDs of only routes from the same AS (excluding confederation sub-ASs). MEDs of two routes are compared only when the first AS number in the AS_Sequence (excluding AS_Confed_Sequence) of one route is the same as its counterpart in the other route.
 - If a route does not carry MED, BGP considers its MED as the default value (0) during route selection. If the **bestroute med-none-as-maximum** command is run, BGP considers its MED as the largest MED value (4294967295).
 - If the **compare-different-as-med** command is run, BGP compares MEDs of routes even when the routes are received from peers in different ASs. If the ASs use the same IGP and route selection mode, you can run this command. Otherwise, do not run this command because a loop may occur.
 - If the **deterministic-med** command is run, routes are no longer selected in the sequence in which they are received.
14. Prefers local VPN routes, LocalCross routes, and RemoteCross routes in descending order.

 NOTE

LocalCross routes refer to locally leaked VPN routes.

If the ERT of a VPNv4 route in the routing table of a VPN instance on a PE matches the IRT of another VPN instance on the PE, the VPNv4 route is added to the routing table of the latter VPN instance. This route is called a LocalCross route. If the ERT of a VPNv4 route learned from a remote PE matches the IRT of a VPN instance on the local PE, the VPNv4 route is added to the routing table of that VPN instance. This route is called a RemoteCross route.

15. Prefers EBGP routes over IBGP routes among the routes learned from peers. In the VPNv4, EVPN, and VPNv6 address families, the routes sent by the local VRF take precedence over the routes learned from peers.
16. Prefers the VPNv4, EVPN, and VPNv6 routes learned from peers.
If the **peer high-priority** command is run, the device preferentially selects the VPNv4, EVPN, and VPNv6 routes learned from IPv4 or IPv6 peers.
17. Prefers the routes that are learned from VPNv4 or VPNv6 peers and are then leaked to a VPN instance and that carry IPv4 or IPv6 next hop addresses.
If the **bestroute nexthop-priority ipv4** command is run, the device preferentially selects the routes that are learned from VPNv4 or VPNv6 peers and are then leaked to a VPN instance and that carry IPv4 next hop addresses.
If the **bestroute nexthop-priority ipv6** command is run, the device preferentially selects the routes that are learned from VPNv4 or VPNv6 peers and are then leaked to a VPN instance and that carry IPv6 next hop addresses.
18. Prefers the route that recurses to an IGP route with the smallest cost.
If the **bestroute igp-metric-ignore** command is run, BGP no longer compares the IGP cost.
19. Prefers the route with the shortest Cluster_List.

 NOTE

By default, Cluster_List takes precedence over Router ID during BGP route selection.
After the **bestroute routerid-prior-clusterlist** command is run, the router ID takes precedence over the cluster list during BGP route selection.

20. Prefers the route advertised by the router with the smallest router ID.
After the **bestroute router-id-ignore** command is run, BGP does not compare router IDs during route selection.

 NOTE

If each route carries an Originator_ID, the originator IDs rather than router IDs are compared during route selection. The route with the smallest Originator_ID is preferred.

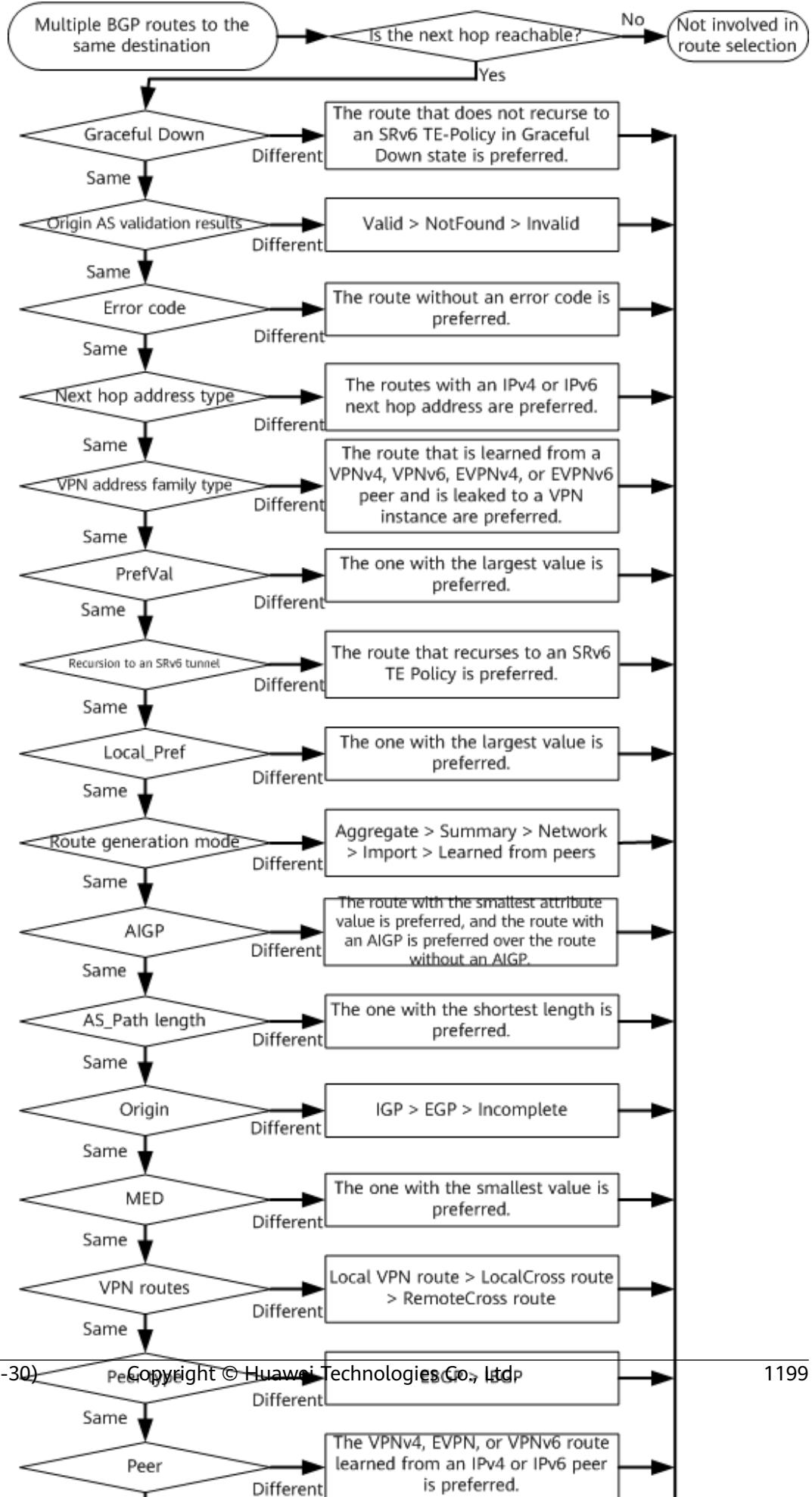
21. Prefers the route learned from the peer with the smallest IP address.
22. If BGP Flow Specification routes are configured locally, the first configured BGP Flow Specification route is preferentially selected.
23. Prefers the locally imported route in the RM routing table.
If a direct route, static route, and IGP route are imported, BGP preferentially selects the direct route, static route, and IGP route in descending order.

24. Prefers the Add-Path route with the smallest recv pathID.
25. Prefers the remotely leaked route with the smallest RD.
26. Prefers locally received routes over the routes imported between VPN and public network instances.
27. Prefers the route that was learned the earliest.

For details about BGP route attributes, see [BGP Attributes](#).

For details about BGP route selection, see [Figure 1-307](#).

Figure 1-307 BGP route selection process



Route Summarization

On a large-scale network, the BGP routing table can be very large. Route summarization can reduce the size of the routing table.

Route summarization is the process of summarizing specific routes with the same IP prefix into a summary route. After route summarization, BGP advertises only the summary route rather than all specific routes to BGP peers.

BGP supports automatic and manual route summarization.

- Automatic route summarization: takes effect on the routes imported by BGP. With automatic route summarization, the specific routes for the summarization are suppressed, and BGP summarizes routes based on the natural network segment and sends only the summary route to BGP peers. For example, 10.1.1.1/32 and 10.2.1.1/32 are summarized into 10.0.0.0/8, which is a Class A address.
- Manual route summarization: takes effect on the local BGP routes. With manual route summarization, users can control the attributes of the summary route and determine whether to advertise the specific routes.

IPv4 supports both automatic and manual route summarization, whereas IPv6 supports only manual route summarization.

BGP Route Advertisement

BGP adopts the following policies to advertise routes:

- When there are multiple valid routes, a BGP speaker advertises only the optimal route to its peers.
- A BGP speaker advertises the routes learned from EBGP peers to all BGP peers, including EBGP peers and IBGP peers.
- A BGP speaker does not advertise the routes learned from an IBGP peer to other IBGP peers.
- Whether a BGP speaker advertises the routes obtained from an IBGP peer to its EBGP peers depends on the BGP-IGP synchronization state.
- A BGP speaker advertises all BGP optimal routes to new peers after peer relationships are established.

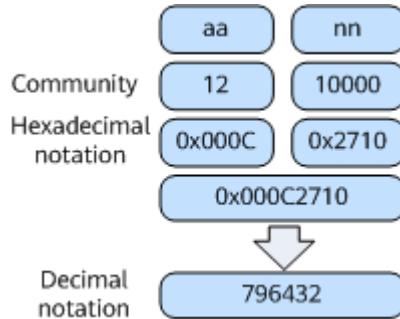
Community Attribute

A community attribute is a route tag used to identify BGP routes with the same characteristics. A community attribute is expressed by a set of 4-byte values. The community attributes on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X are expressed in two formats: hexadecimal format (aa:nn) and decimal integer format (community number). The two formats can be converted to each other, as shown in [Figure 1-308](#).

- aa:nn: aa indicates an AS number and nn indicates the community identifier defined by an administrator. The value of aa or nn ranges from 0 to 65535, which is configurable. For example, if a route is from AS 100 and the community identifier defined by the administrator is 1, the community is 100:1.
- Community number: It is an integer ranging from 0 to 4294967295. As defined in standard protocols, numbers from 0 (0x00000000) to 65535

(0x0000FFFF) and from 4294901760 (0xFFFF0000) to 4294967295 (0xFFFFFFFF) are reserved.

Figure 1-308 Mapping between the two community attribute formats



Community attributes simplify the application, maintenance, and management of route-policies and allow a group of BGP devices in multiple ASs to share a route-policy. Community attributes are route attributes transmitted between BGP peers, free from the restriction by the AS. A BGP device can be configured to change the original community attribute of a route before advertising the route to peers.

The peers in a peer group share the same policy, while the routes with the same community attribute share the same policy.

In addition to well-known community attributes, you can use a community filter to define extended community attributes to flexibly control route-policies.

Well-known Community Attributes

Table 1-103 lists well-known BGP community attributes.

Table 1-103 Well-known BGP community attributes

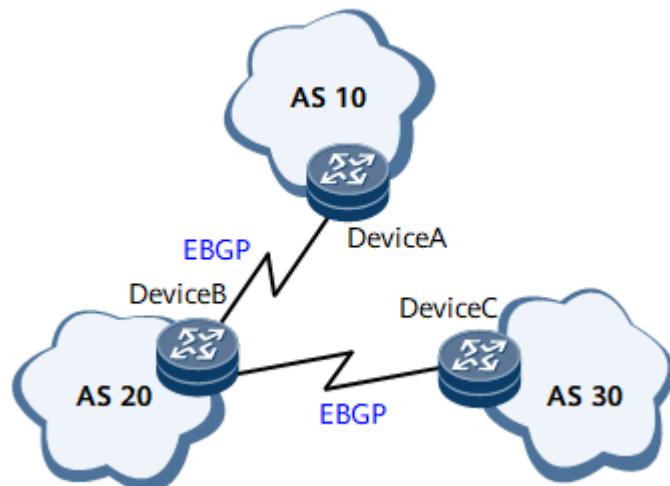
Community Name	Community Identifier	Description
Internet	0 (0x00000000)	By default, all routes belong to the Internet community. A route with this attribute can be advertised to all BGP peers.
No_Export	4294967041 (0xFFFFF01)	After being received, a route with this attribute cannot be advertised outside the local AS.
No_Advertise	4294967042 (0xFFFFF02)	After being received, a route with this attribute cannot be advertised to any other BGP peers.
No_Export_Subconfed	4294967043 (0xFFFFF03)	After being received, a route with this attribute cannot be advertised outside the local AS or to other sub-ASs.

Community Name	Community Identifier	Description
GRACEFUL_SHUT DOWN	4294901760 (0xFFFF0000)	During graceful shutdown, routes with this attribute are advertised, notifying other devices that the local device is in the graceful shutdown state.

Usage Scenario

On the network shown in [Figure 1-309](#), EBGP connections are established between DeviceA and DeviceB, and between DeviceB and DeviceC. If the No_Export community attribute is configured on DeviceA in AS 10 and DeviceA sends a route with the community attribute to DeviceB in AS 20, DeviceB does not advertise the route to other ASs after receiving it.

Figure 1-309 Networking for BGP communities



Large-Community Attribute

A BGP community attribute is a set of destination addresses that have the same characteristics. The attribute is expressed as a list of one or more 4-byte values. Generally, the community attribute on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X is in the format of aa:nn, where aa indicates an Autonomous System Number (ASN) and nn indicates an attribute ID defined by the network administrator. Due to the use of 4-byte ASNs, the BGP community attribute can no longer accommodate both an ASN and attribute ID. In addition, the community attribute offers limited flexibility because only one attribute ID is available. To address these shortcomings, the Large-Community attribute is defined. This attribute extends and can be used together with the community attribute. The Large-Community attribute is a set of one or more 12-byte values, each of which is in the format of Global Administrator:LocalData1:LocalData2.

The Global Administrator field is intended to represent a complete 4-byte ASN, but it can also be set to a value other than an ASN. Global Administrator,

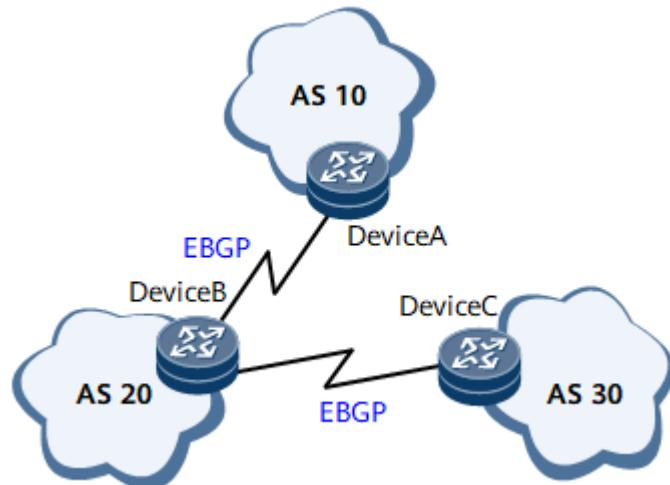
LocalData1, and LocalData2 are unsigned integers each ranging from 0 to 4294967295. The administrator can set Global Administrator, LocalData1, and LocalData2 according to requirements.

The Large-Community attribute can represent a 2-byte or 4-byte ASN, and has two 4-byte LocalData IDs. This allows an administrator to apply route-policies more flexibly. For example, the Large-Community attribute can be set to ME:ACTION:YOU or ASN:Function:Parameter.

Networking Application

On the network shown in [Figure 1-310](#), Device B establishes an EBGP peer relationship with each of Device A and Device C. By setting the Large-Community attribute to 20:4:30 on Device B, you can disable the routes on Device A from being advertised to AS30.

Figure 1-310 Networking diagram for configuring the BGP Large-Community attribute



AIGP

Background

The Accumulated Interior Gateway Protocol Metric (AIGP) attribute is an optional non-transitive Border Gateway Protocol (BGP) path attribute. The attribute type code assigned by the Internet Assigned Numbers Authority (IANA) for the AIGP attribute is 26.

Routing protocols that run within a single administrative domain, such as IGP, assign a metric (also called a cost) to each link and select the path with the smallest metric. BGP, designed to provide routing over a large number of independent administrative domains, does not select paths based on metrics. If a single administrative domain (AIGP domain) consists of several BGP networks, it is desirable for BGP to select paths based on metrics, just as an IGP does. The AIGP attribute enables BGP to select paths based on metrics.

Related Concepts

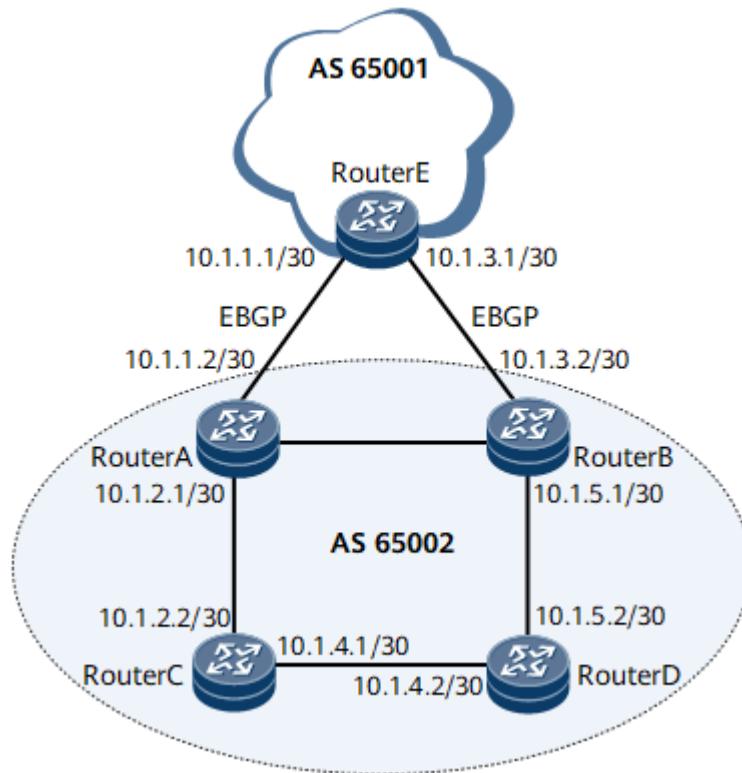
An AIGP administrative domain is a set of autonomous systems (ASes) in a common administrative domain. The AIGP attribute takes effect only in an AIGP administrative domain.

Implementation

AIGP Attribute Origination

The AIGP attribute can be added to a route only through a route-policy. You can configure a route-policy to set an AIGP value for the routes imported, received, or sent by BGP. If no AIGP value is configured, the IGP routes imported by BGP do not carry the AIGP attribute. [Figure 1-311](#) shows the typical AIGP application networking.

Figure 1-311 AIGP application networking



AIGP Attribute Transmission

BGP cannot transmit the AIGP attribute outside the AIGP domain. If the AIGP attribute of a route changes, BGP sends Update packets for BGP peers to update information about this route. In a scenario in which A, a BGP speaker, sends a route that carries the AIGP attribute to B, its BGP peer:

- If B does not support the AIGP attribute or does not have the AIGP capability enabled for a peer, B ignores the AIGP attribute and does not transmit the AIGP attribute to other BGP peers.
- If B supports the AIGP attribute and has the AIGP capability enabled for a peer, B can modify the AIGP attribute of the route only after B has set itself to

be the next hop of the route. The rules for modifying the AIGP attribute are as follows:

- If the BGP peer relationship between A and B is established over an IGP route, or a static route that does not require recursion, B uses the metric value of the IGP or static route plus the received AIGP attribute value as the new AIGP attribute value and sends the new AIGP attribute to other peers.
- If the BGP peer relationship between A and B is established over a BGP route, or a static route that requires recursion, route recursion is performed when B sends data to A. Each route recursion involves a recursive route. B uses the sum of the metric values of the recursive routes plus the received AIGP attribute value as the new AIGP attribute value and sends the new AIGP attribute to other peers.

Role of the AIGP Attribute in BGP Route Selection

When there are multiple active routes to the same destination, BGP performs route selection. If BGP fails to select the optimal route after comparing PrefVal, Local_Pref, and route-type information, BGP compares the AIGP attributes of the routes. The rules are as follows:

- AIGP comparison takes place after route-type comparison and before AS_Path comparison.
- The priority of a route that carries the AIGP attribute is higher than the priority of a route that does not carry the AIGP attribute.
- If all routes carry the AIGP attribute, the route with the smallest AIGP attribute value plus the IGP metric value of the recursive next hop is preferred over the other routes.

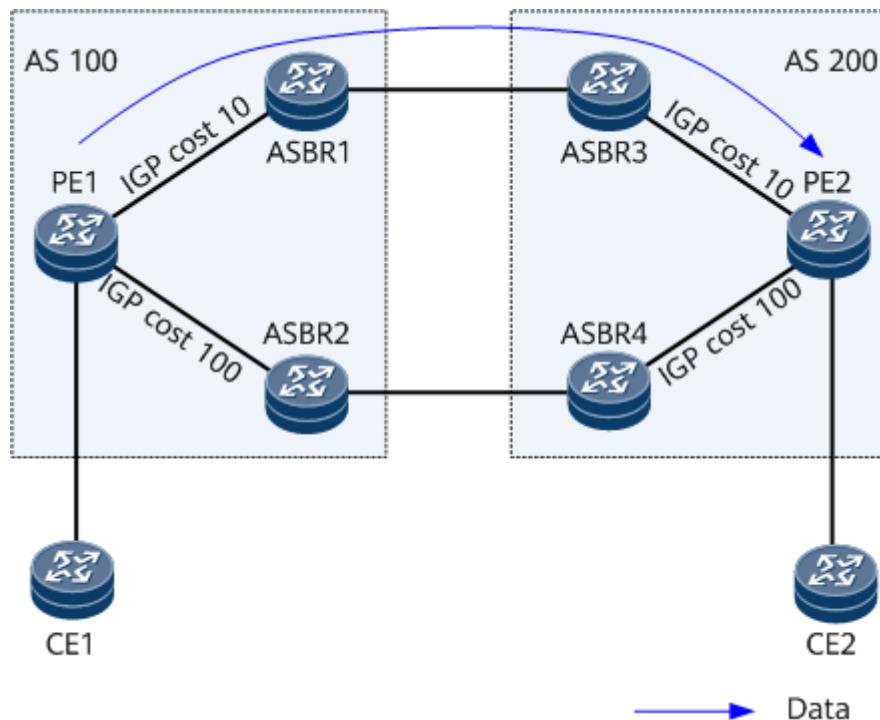
Usage Scenario

The AIGP attribute is used to select the optimal route in an AIGP administrative domain.

The AIGP attribute can be transmitted between BGP unicast peers as well as between BGP VPNv4/VPNv6 peers. Transmitting the AIGP attribute between BGP VPNv4/VPNv6 peers allows L3VPN traffic to be transmitted along the path with the smallest AIGP attribute value.

On the inter-AS IPv4 VPN Option B network shown in [Figure 1-312](#), BGP VPNv4 peer relationships are established between PEs and ASBRs. Two paths with different IGP costs exist between PE1 and PE2. If you want the PEs to select a path with a lower IGP cost to carry traffic, you can enable the AIGP capability in the BGP VPNv4 address family view and configure a route-policy to add the same AIGP initial value to BGP VPNv4 routes. Take PE1 as an example. After this configuration, PE1 receives two BGP VPNv4 routes destined for CE2 from ASBR1 and ASBR2, and the BGP VPNv4 route sent by ASBR1 has a lower AIGP value. If higher-priority route selection conditions of the routes are the same, PE1 preferentially selects the route with a lower AIGP value, and traffic is transmitted over the PE1 -> ASBR1 -> ASBR3 -> PE2 path.

Figure 1-312 Applying AIGP in inter-AS IPv4 VPN Option B scenarios



Benefits

After the AIGP attribute is configured in an AIGP administrative domain, BGP selects optimal routes based on route metrics, just as an IGP does. This ensures that all devices in the AIGP administrative domain forward data along the optimal paths.

Entropy Label

BGP entropy labels comply with related standards and cannot be used for forwarding. In addition, BGP entropy labels are not allocated through protocol negotiation. The only function of BGP entropy labels is to improve the hash effect of load balancing. An entropy label can be generated through BGP negotiation to improve load balancing during traffic forwarding. A well-known entropy label (label 7) is added to the inner layer of the BGP LSP label, and then a random label is added to the inner layer of the BGP LSP label.

Background

As user networks and the scope of network services continue to expand, load-balancing techniques are used to improve bandwidth between nodes. If tunnels are used for load balancing, transit nodes (P) obtain IP content carried in MPLS packets as a hash key. If a transit node cannot obtain the IP content from MPLS packets, the transit node can only use the top label in the MPLS label stack as a hash key. The top label in the MPLS label stack cannot differentiate underlying-layer protocols in packets in detail. As a result, the top MPLS labels are not distinguished when being used as hash keys, resulting in load imbalance. Per-packet load balancing can be used to prevent load imbalance but results in disorder. This drawback adversely affects user experience. The entropy label

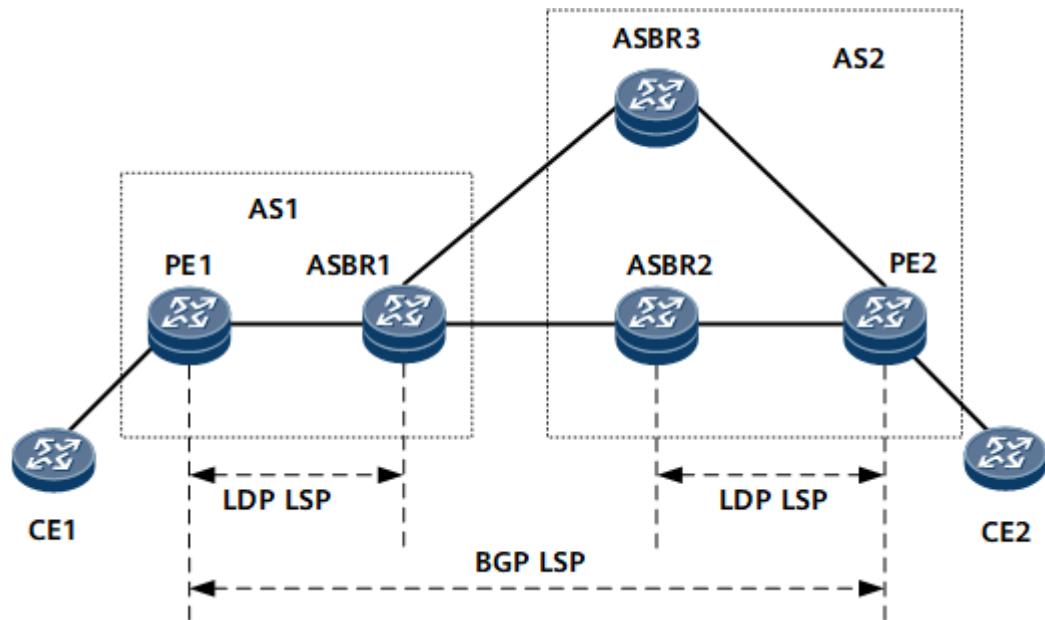
feature can solve the preceding problems and improve load balancing performance.

Implementation

On the network shown in [Figure 1-313](#), load balancing is performed on ASBRs (transit nodes) and the result is uneven. To achieve even load balancing, you can configure the entropy label capability of the BGP LSP.

An entropy label is generated by an ingress label switching router (LSR) based on load balancing information. To help the egress LSR distinguish the entropy label generated by the ingress LSR from application labels, label 7 is added before an entropy label in the MPLS label stack.

Figure 1-313 Load balancing performed among transit nodes



The ingress generates an entropy label and encapsulates it into the MPLS label stack. Before the ingress LSR encapsulates packets with MPLS labels, it can easily obtain IP or Layer 2 protocol data for use as a hash key. If the ingress detects the entropy label capability enabled for tunnels, it uses the IP information carried in packets to compute an entropy label, adds the label to the MPLS label stack, and advertises the label to an ASBR. The ASBR uses the entropy label as a hash key to load-balance traffic and does not need to parse IP data inside the packets.

The entropy label is pushed into packets by the ingress and removed by the egress. Therefore, the egress needs to notify the ingress of the support for the entropy label capability.

Each node in [Figure 1-313](#) processes the entropy label as follows:

- Egress (PE2): If the egress can parse entropy labels, it adds the entropy label attribute to BGP routes and then advertises the routes to the upstream node to notify the upstream node of its support of the entropy label capability. The routes are then transmitted hop by hop until to the ingress.
- Transit node (ASBR): A transit node needs to be enabled with the entropy label attribute advertisement capability so that the transit node can advertise

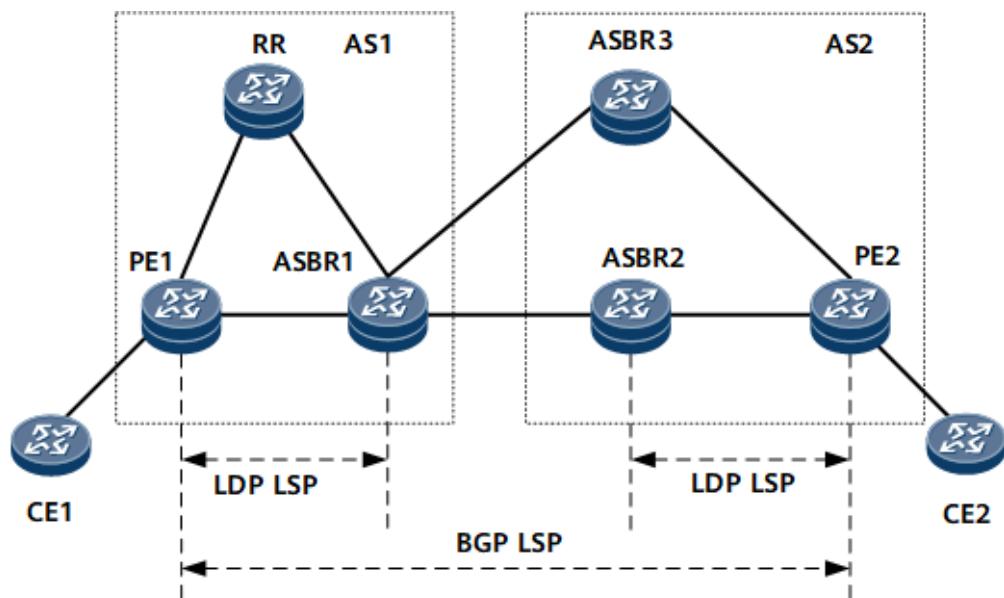
BGP routes to notify the upstream node of its support of the entropy label capability.

- Ingress (PE1): The ingress determines whether to add an entropy label to improve load balancing based on the entropy label capability advertised by the egress.

Usage Scenario

- In [Figure 1-313](#), entropy labels are used when load balancing is performed among transit nodes.
- On the network shown in [Figure 1-314](#), the BGP labeled routes exchanged between PE1 and ASBR1 are sent through an RR. If the RR needs to advertise the entropy label attribute, BGP LSP Entropy label attribute advertisement needs to be enabled between the RR and PE1, and between the RR and ASBR1. The RR also needs to be enabled to forward BGP LSP entropy labels. If the RR is not enabled to forward BGP LSP entropy labels, it discards the BGP LSP entropy labels carried in routes.

Figure 1-314 BGP LSP RR scenario



Benefits

The entropy label can improve load balancing.

BGP Routing Loop Detection

Once a routing loop occurs on a Layer 3 network, packets cannot be forwarded, which may cause losses to carriers or users. To detect potential routing loops on the network, BGP defines the Loop-detection attribute.

Fundamentals of BGP Routing Loop Detection

The fundamentals of BGP routing loop detection are as follows:

1. After BGP routing loop detection is enabled, the local device generates a random number, adds the Loop-detection attribute to the routes to be advertised to EBGP peers or the locally imported routes to be advertised to peers, and encapsulates the attribute with the random number and the local **vrfID**. The local **vrfID** is automatically generated and globally unique. In the public network scenario, the **vrfID** is 0. In the private network scenario, the **vrfID** is automatically generated after a VPN instance is created. When OSPF/IS-IS routes are imported to BGP, the routing loop attributes of the OSPF/IS-IS routes are inherited.
2. When the local device receives a route with the Loop-detection attribute from another device, the local device performs the following checks:
 - Compares the Loop-detection attribute of the received route with the combination of the **vrfID** and random number that are locally stored.
 - If they are the same, the local device determines that a routing loop occurs.
 - If they are different, the local device determines that no routing loop occurs, and the route participates in route selection.
 - Checks whether the received route has a routing loop record.

 NOTE

Routing loop records are affected by the following commands:

- For the routes that already have routing loop records before routing loop alarms are cleared using the **clear route loop-detect bgp alarm** command, the records always exist.
 - For the routes that have routing loop records but no longer carry the routing loop attribute of the local device after routing loop alarms are cleared using the **clear route loop-detect bgp alarm** command, the records of these routes are deleted.
 - If the **undo route loop-detect bgp enable** command is run, the routing loop records of all looped routes are deleted.
 - If a route has a routing loop record, a routing loop once occurred. Such a route is considered to be a looped route even if it does not carry the routing loop attribute of the local device.
 - If there is no routing loop record and the Loop-detection attribute of the received route is different from the combination of the **vrfID** and random number that are locally stored, the local device determines that no routing loop occurs, and the route participates in route selection normally.
 - If there is no routing loop record but the Loop-detection attribute of the received route is the same as the combination of the **vrfID** and random number that are locally stored, the local device determines that a routing loop occurs.
3. If a routing loop is detected and the looped route is selected, the local device reports an alarm to notify the user of the routing loop risk, enters the loop prevention state, and processes the looped route.
 - Preferentially selects non-looped routes when the BGP routing table contains multiple routes with the same destination as the looped route.

- Increases the MED value and reduces the local preference of the looped route when advertising it.
4. After the device processes the looped route, the routing loop may be resolved. If the routing loop persists, you need to locate the cause of the loop and resolve the loop. As the device cannot detect when the loop risk is eliminated, the routing loop alarm will not be cleared automatically. To manually clear the alarm after the loop risk is eliminated, you can run a related command.

Implementation

The Loop-detection attribute is a private BGP attribute. It uses a reserved value (type=255) to implement routing loop detection in some scenarios. **Figure 1-315** shows the Loop-detection attribute TLV, and **Table 1-104** describes the fields in it.

NOTE

Currently, the Loop-detection attribute is supported only in the BGP IPv4 public network, BGP IPv4 VPN, BGP IPv6 public network, BGP IPv6 VPN, BGP VPNv4, and BGP VPNv6 address families. When a Huawei device interworks with a non-Huawei device, check whether the non-Huawei device can process Type 255 messages before enabling routing loop detection.

Figure 1-315 Loop-detection attribute TLV extension

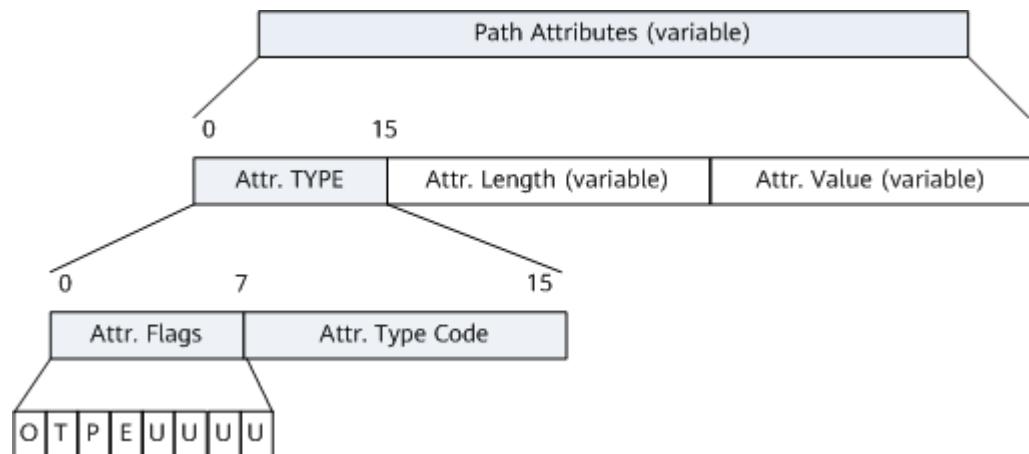


Table 1-104 Fields in the Loop-detection attribute TLV

Field	Description
Attr.Flags	Attribute flag, which occupies one byte (eight bits). The meaning of each bit is as follows: O (Optional bit): defines whether the attribute is optional. The value 1 indicates an optional attribute, whereas the value 0 indicates a well-known attribute. T (Transitive bit): Defines whether the attribute is transitive. For an optional attribute, the value 1 indicates that the attribute is transitive, whereas the value 0 indicates that the attribute is non-transitive. For a well-known attribute, the value must be set to 1. P (Partial bit): defines whether the attribute is partial. If the optional transitive attribute is partial, the value is set to 1; if the attribute is complete, the value is set to 0. For well-known attributes and for optional non-transitive attributes, the value must be set to 0. E (Extended Length bit): defines whether the length (Attr. Length) of the attribute needs to be extended. If the attribute length does not need to be extended, the value is set to 0 and the Attr. Length is 1 octet. If the attribute length needs to be extended, the value is set to 1 and the Attr. Length is 2 octets. U (Unused bits): Indicates that the lower-order 4 bits are not used. These bits must be set to 0s upon transmission and ignored upon receipt.
Attr.Type Code	Attribute type, which occupies one byte. The value is an unsigned integer, with the initial value being 0xFF.
Attr.Length	Length of the attribute.
Attr.Value	The value is 0x0030FBB8. It is Huawei's organizationally unique identifier (OUI) and is used for vendor identification.

BGP also defines a sub-TLV for Attr.Value to identify the device that detects a routing loop. [Figure 1-316](#) shows the sub-TLV, and [Table 1-105](#) describes the fields in the sub-TLV.

 **NOTE**

A maximum of four Loop-detection attribute sub-TLVs can be carried. If more than four sub-TLVs exist, they are overwritten according to the first-in-first-out rule.

Figure 1-316 Loop-detection attribute sub-TLV

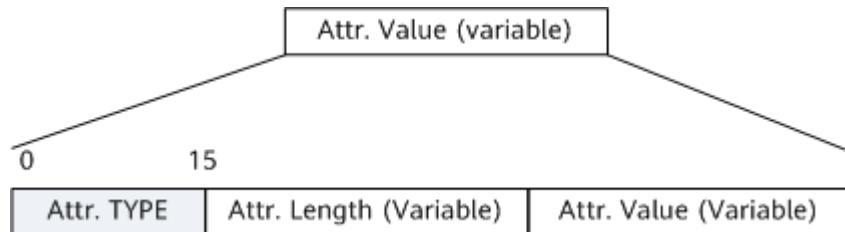


Table 1-105 Fields in the Loop-detection attribute sub-TLV

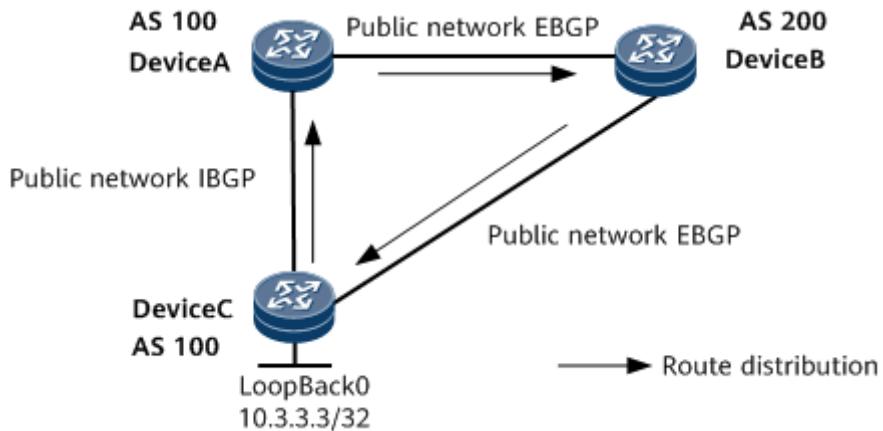
Field	Description												
Attr.Type	The value is 0xFF.												
Attr.Length	Length of the attribute. The value is 0x08, 0x10, 0x18, or 0x20.												
Attr.Value	<table border="1"><tr><td>0</td><td>31</td><td>63</td></tr><tr><td>+-----+</td><td>+-----+</td><td>+-----+</td></tr><tr><td> vrfID Random number </td><td></td><td></td></tr><tr><td>+-----+</td><td>+-----+</td><td>+-----+</td></tr></table> <p>vrfID specifies a system-allocated VPN ID. The value ranges from 0 to 0xFFFFFFFF.</p> <p>NOTE For BGP VPNv4 and BGP VPNv6 routes, the system only checks the random number when determining whether a routing loop occurs; that is, it does not check the vrfID.</p>	0	31	63	+-----+	+-----+	+-----+	vrfID Random number			+-----+	+-----+	+-----+
0	31	63											
+-----+	+-----+	+-----+											
vrfID Random number													
+-----+	+-----+	+-----+											

Usage Scenario

BGP routing loops may occur in the following scenarios. You are advised to enable BGP routing loop detection during network planning.

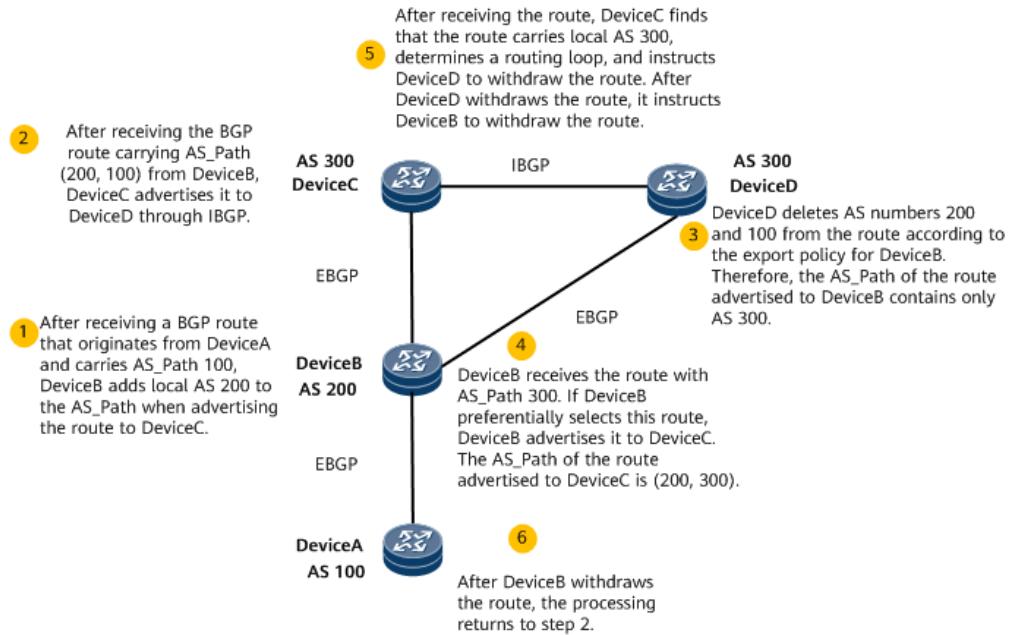
- On the network shown in [Figure 1-317](#), DeviceA and DeviceC belong to AS 100, and DeviceB belongs to AS 200. An export policy is configured on DeviceB to delete the original AS numbers from the routes to be advertised to DeviceC. After receiving a BGP route that originates from DeviceC, DeviceA advertises the route to DeviceB, which then advertises the route back to DeviceC. As a result, a BGP routing loop occurs on DeviceC. After BGP routing loop detection is enabled on the entire network, DeviceC adds Loop-detection attribute 1 to the BGP route (locally imported) before advertising the route to DeviceA. After receiving the route, DeviceA adds Loop-detection attribute 2 to the route before advertising the route to DeviceB (EBGP peer). After receiving the route, DeviceB adds Loop-detection attribute 3 to the route before advertising the route to DeviceC (EBGP peer). After receiving the Loop-detection attributes, DeviceC discovers that these attributes contain Loop-detection attribute 1 which was added by itself, and then reports a routing loop alarm.

Figure 1-317 Typical networking 1 with a BGP routing loop



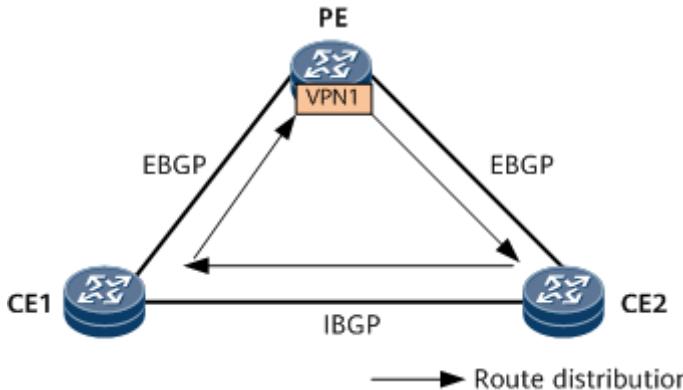
- On the network shown in [Figure 1-318](#), DeviceA resides in AS 100; DeviceB resides in AS 200; DeviceC and DeviceD reside in AS 300. An export policy is configured on DeviceD to delete the original AS numbers from the routes to be advertised to DeviceB. In this scenario, a BGP routing loop occurs on DeviceB.

Figure 1-318 Typical networking 2 with a BGP routing loop



- On the network shown in [Figure 1-319](#), the PE advertises a VPN route through VPN1, and then receives this route through VPN1, indicating that a routing loop occurs on the PE.

Figure 1-319 Typical networking 3 with a BGP routing loop

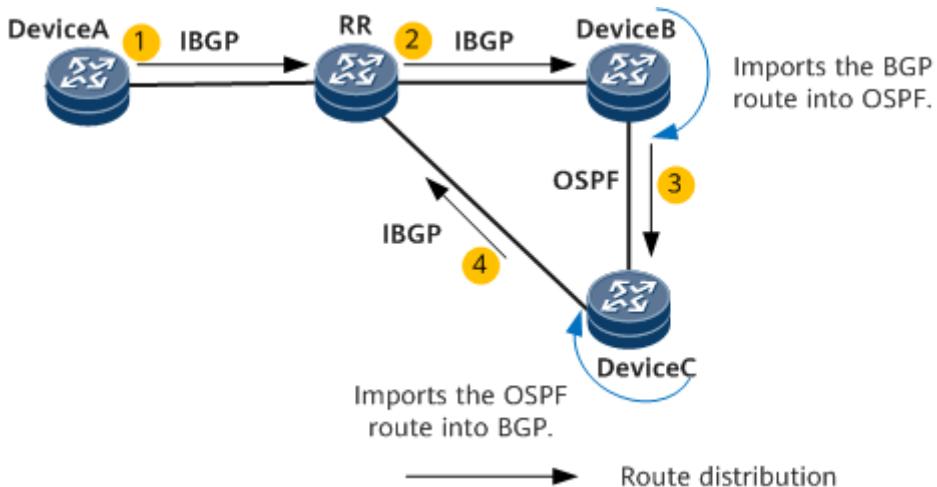


- On the network shown in [Figure 1-320](#), DeviceA, DeviceB, and DeviceC belong to AS 100. An IBGP peer relationship is established between DeviceA and the RR, between the RR and DeviceB, and between the RR and DeviceC. OSPF runs on DeviceB and DeviceC. DeviceB is configured to import BGP routes to OSPF, and DeviceC is configured to import OSPF routes to BGP. An export policy is configured on DeviceA to add AS numbers to the AS_Path attribute for the routes to be advertised to the RR. After receiving a BGP route from DeviceA, the RR advertises this route to DeviceB. DeviceB then imports the BGP route to convert it to an OSPF route and advertises the OSPF route to DeviceC. DeviceC then imports the OSPF route to convert it to a BGP route and advertises the BGP route to the RR. When comparing the route advertised by DeviceA and the route advertised by DeviceC, the RR prefers the one advertised by DeviceC as it has a shorter AS_Path than that of the route advertised by DeviceA. As a result, a stable routing loop occurs.

To address this problem, enable BGP routing loop detection on DeviceC. After BGP routing loop detection is enabled, DeviceC adds Loop-detection attribute 1 to the BGP route imported from OSPF and advertises the BGP route to the RR. After receiving this BGP route, the RR advertises it (carrying Loop-detection attribute 1) to DeviceB. As OSPF routing loop detection is enabled by default, when the BGP route is imported to become an OSPF route on DeviceB, the OSPF route inherits the routing loop attribute of the BGP route and has an OSPF routing loop attribute added as well before the OSPF route is advertised to DeviceC. Upon receipt of the OSPF route, DeviceC imports it to convert it to a BGP route. Because BGP routing loop detection is enabled, the BGP route inherits the routing loop attributes of the OSPF route. Upon receipt of the route, DeviceC finds that the received route carries its own routing loop attribute and therefore determines that a routing loop has occurred. In this case, DeviceC generates an alarm, and reduces the local preference and increases the MED value of the route before advertising the route to the RR. After receiving the route, the RR compares this route with the route advertised by DeviceA. Because the route advertised by DeviceC has a lower local preference and a larger MED value, the RR preferentially selects the route advertised by DeviceA. The routing loop is then resolved.

When the OSPF route is transmitted to DeviceC again, DeviceC imports it to convert it to a BGP route, and the route carries only the OSPF routing loop attribute added by DeviceB. However, DeviceC still considers the route as a looped route because the route has a routing loop record. In this case, the RR does not preferentially select the route after receiving it from DeviceC. Then routes converge normally.

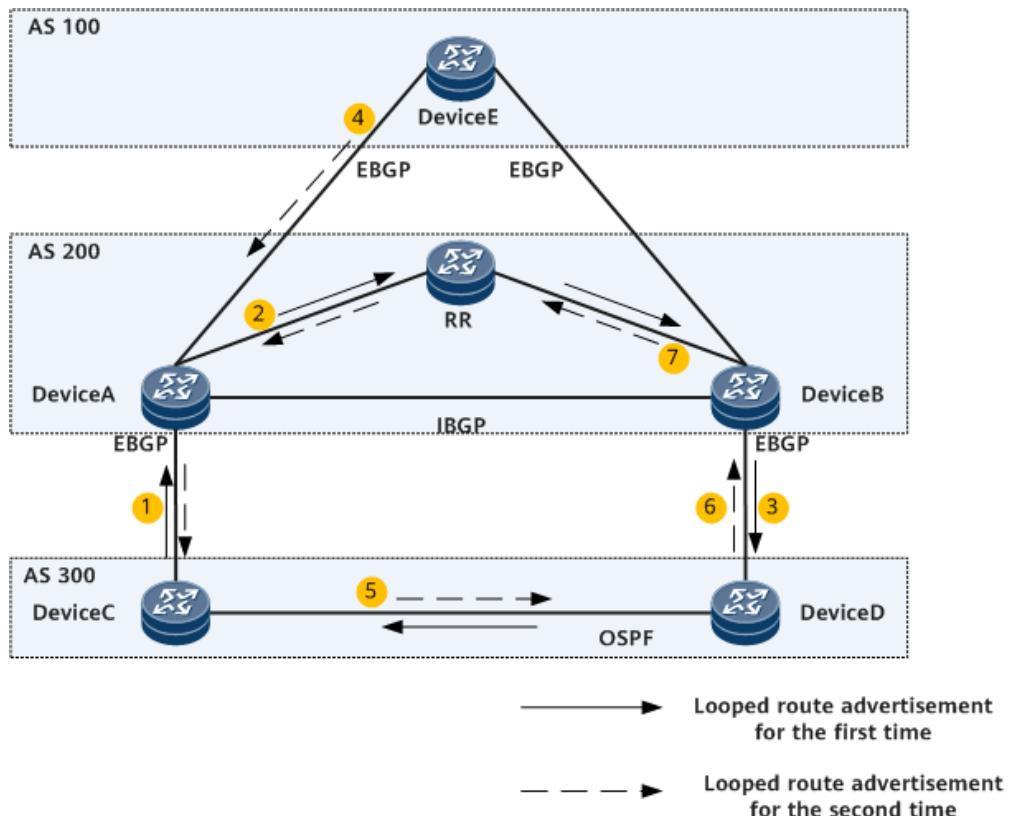
Figure 1-320 Typical networking 4 with a BGP routing loop



- As shown in [Figure 1-321](#), EBGP peer relationships are established between DeviceA and DeviceC, between DeviceA and DeviceE, between DeviceB and DeviceD, and between DeviceB and DeviceE. IBGP peer relationships are established between DeviceA and the RR, between DeviceB and the RR, and between DeviceA and DeviceB. An OSPF neighbor relationship is established between DeviceC and DeviceD.
 - a. In normal situations, DeviceA learns an EBGP route from DeviceE, and the AS_Path of the route is (100, 200). After an EBGP peer relationship is established between DeviceA and DeviceC, assume that DeviceA advertises this route to DeviceC first. According to BGP route selection rules, locally imported routes are preferentially selected. In this case, DeviceC preferentially selects the OSPF route that is imported into BGP. DeviceC then advertises the route to DeviceA. The AS_Path carried in the route is DeviceC's AS number (300). After receiving this route from DeviceC, DeviceA preferentially selects this route because the AS_Path of this route is shorter.
 - b. After selecting this optimal route, DeviceA advertises it to the RR through the VPNv4 peer relationship. The RR then reflects the route to DeviceB.
 - c. DeviceB selects this route based on the AS_Path attribute and advertises this route to DeviceD. The AS_Path attribute carried in the route is DeviceC's AS number (300). Because DeviceD and DeviceC have the same AS number, DeviceD detects an AS_Path loop and deletes this route. Then, DeviceD withdraws the OSPF route locally and advertises an OSPF route withdrawal message to DeviceC. After receiving the OSPF route withdrawal message, DeviceC deletes the route imported into BGP and advertises a BGP route withdrawal message to DeviceA.
 - d. After receiving the BGP route withdrawal message from DeviceC, DeviceA re-selects the route advertised by DeviceE and advertises this route to DeviceC.
 - e. After receiving the BGP route from DeviceA, DeviceC imports the route into OSPF and advertises the OSPF route to DeviceD through the OSPF neighbor relationship.
 - f. After receiving the OSPF route, DeviceD imports it into BGP and advertises the BGP route to DeviceB.

- g. After receiving the BGP route from DeviceD, DeviceB preferentially selects this route because the AS_Path attribute of this route is shorter. Then, DeviceB advertises this route to the RR through the VPv4 peer relationship. As a result, a routing loop occurs again.

Figure 1-321 Typical networking 5 with a BGP routing loop



NOTE

This function is not supported in the following scenarios:

- When BGP is configured to advertise default routes, the Loop-detection attribute is not added to the generated default routes but is added to the imported default routes.
- When BGP Add-Path is configured, the Loop-detection attribute is not added to routes.
- When the route server function is configured, the Loop-detection attribute is not added to routes.
- After receiving a route from a peer and then advertising the route to an IBGP peer, the device does not add its own Loop-detection attribute to the route. That is, the route carries only the original Loop-detection attribute.

BGP Peer Group and Dynamic Peer Group

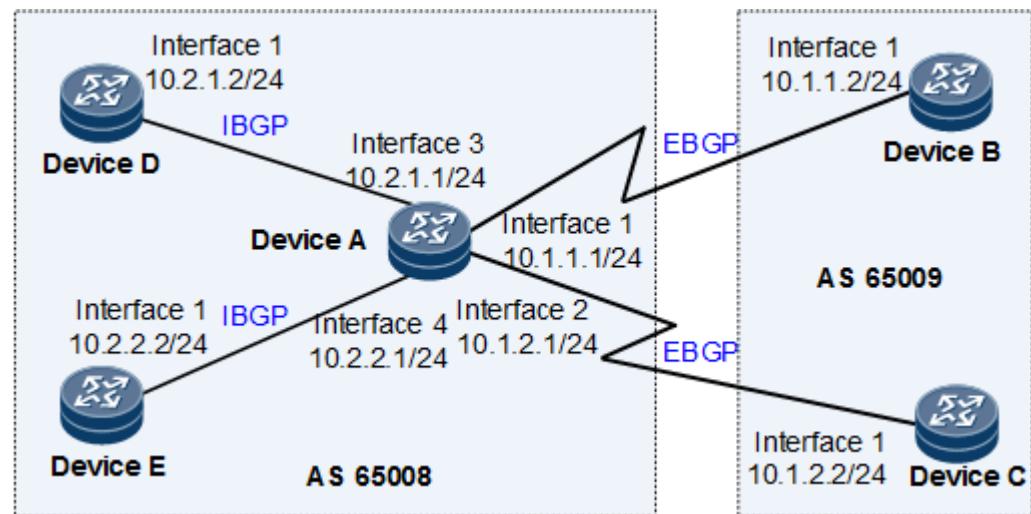
A peer group is a set of peers with the same policies. After a peer is added to a peer group, the peer inherits the configurations of this peer group. If the configurations of the peer group change, the configurations of all the peers in the group change accordingly. A large number of BGP peers may exist on a large-scale BGP network. If many of the BGP peers need the same policies, some commands need to be run repeatedly for each peer. To simplify the configuration, you can configure a peer group. Each peer in a peer group can be configured with unique policies to advertise and receive routes.

However, multiple BGP peers can change frequently on some BGP networks, causing the establishment of BGP peer relationships to change accordingly. If you configure peers in static mode, you must frequently add or delete peer configurations on the local device, which increases the maintenance workload. To address this problem, configure the dynamic BGP peer function to enable BGP to listen for BGP connection requests from a specified network segment, dynamically establish BGP peer relationships, and add these peers to the same dynamic peer group. This spares you from adding or deleting BGP peer configurations in response to each change in dynamic peers, reducing the workload of network maintenance.

Application

On the network shown in [Figure 1-322](#), an EBGP peer relationship is established between Device A and Device B and between Device A and Device C, and an IBGP peer relationship is established between Device A and Device D and between Device A and Device E.

Figure 1-322 Dynamic BGP peer groups

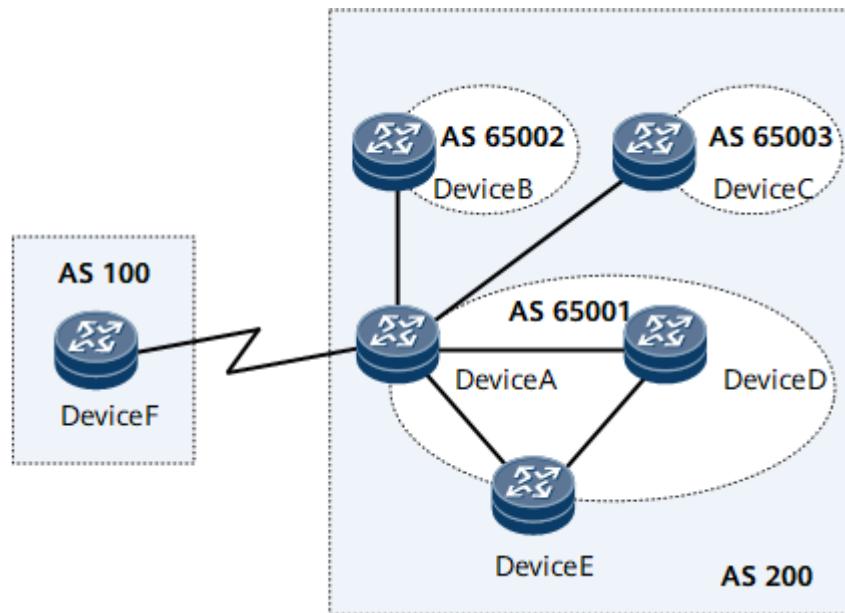


Device B and Device C are on the same network segment (10.1.0.0/16). In this case, you can configure a dynamic peer group on Device A to listen for BGP connection requests from this network segment. After the dynamic peer group is configured, Device B and Device C are dynamically added to this peer group, and the devices to be deployed on this network segment will also be dynamically added to the peer group when they request to establish BGP peer relationships with Device A. This process helps reduce the network maintenance workload. In addition, you can configure another dynamic peer group on Device A so that Device D and Device E are dynamically added to this peer group.

BGP Confederation

Confederation is a method of handling the surge of IBGP connections in an AS. It divides an AS into several sub-ASs. IBGP full-mesh connections are established in each sub-AS, and EBGP connections are established between sub-ASs, as shown in [Figure 1-323](#).

Figure 1-323 Confederation



For a BGP speaker (for example, the router in AS 100) that does not belong to a confederation, multiple sub-ASs (AS 65001, AS 65002, and AS 65003) in the same confederation are considered as a whole. External devices do not need to know the status of internal sub-ASs. The confederation ID, which identifies the confederation, is set to the AS number. As shown in [Figure 1-323](#), AS 200 is used as a confederation ID.

In [Figure 1-323](#), multiple BGP devices reside in AS 200. To reduce the number of IBGP connections to be established, three sub-ASs: AS 65001, AS 65002, and AS 65003 are deployed. In AS 65001, fully meshed IBGP connections are established between the three routers.

Applications and Limitations

The confederation needs to be configured on each router, and the routers that join a confederation must have the confederation function.

The confederation has the following disadvantage: When the non-confederation solution is changed to the confederation solution, the routers need to be reconfigured, and the logical topology needs to be changed.

On large-scale BGP networks, both the RR and confederation solutions can be used.

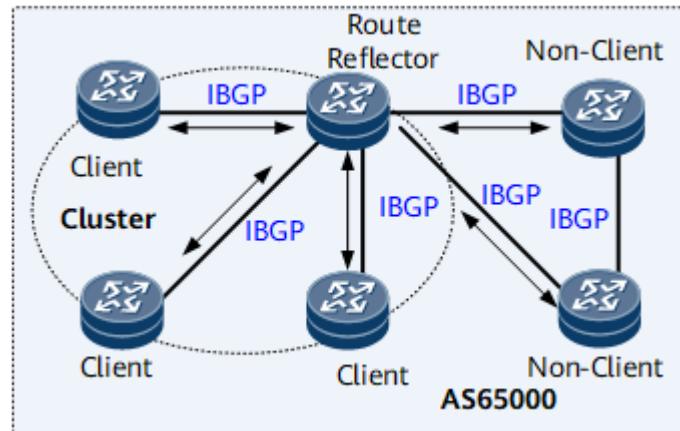
Route Reflector

Fully meshed connections need to be established between IBGP peers to ensure the connectivity between IBGP peers. If there are n routers in an AS, $n \times (n-1)/2$ IBGP connections need to be established. When there are a lot of IBGP peers, network resources and CPU resources are greatly consumed. Route reflection can solve the problem.

In an AS shown in [Figure 1-324](#), one router functions as a Route Reflector (RR), with some other routers as its clients. The clients establish IBGP connections with the RR. The RR and its clients form a cluster. The RR reflects routes among clients, and BGP connections do not need to be established between the clients.

A BGP peer that functions as neither an RR nor a client is called a non-client. A non-client must establish fully meshed connections with the RR and all the other non-clients.

Figure 1-324 Networking with an RR



Application

After receiving routes from peers, an RR selects the optimal route based on BGP route selection rules and advertises the optimal route to other peers based on the following rules:

- If the optimal route is from a non-client IBGP peer, the RR advertises the route to all clients.
- If the optimal route is from a client, the RR advertises the route to all non-clients and clients.
- If the optimal route is from an EBGP peer, the RR advertises the route to all clients and non-clients.

An RR is easy to configure because it only needs to be configured on the router that needs to function as an RR, and clients do not need to know whether they are clients.

On some networks, if fully meshed connections have already been established among clients of an RR, they can exchange routing information directly. In this case, route reflection among the clients through the RR is unnecessary and occupies bandwidth. For example, on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, route reflection through the RR can be disabled, but the routes between clients and non-clients can still be reflected. By default, route reflection between clients through the RR is enabled.

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, an RR can change various attributes of BGP routes, such as the AS_Path, MED, Local_Pref, and community attributes.

Originator_ID

Originator_ID and Cluster_List are used to detect and prevent routing loops.

The Originator_ID attribute is four bytes long and is generated by an RR. It carries the router ID of the route originator in the local AS.

- When a route is reflected by an RR for the first time, the RR adds the Originator_ID attribute to this route. The Originator_ID attribute is used to identify the router that originates the route. If a route already carries the Originator_ID attribute, the RR does not create a new one.
- After receiving the route, a BGP speaker checks whether the Originator_ID is the same as its router ID. If Originator_ID is the same as its router ID, the BGP speaker discards this route.

Cluster_List

To prevent routing loops between ASs, a BGP router uses the AS_Path attribute to record the ASs through which a route passes. Routes with the local AS number are discarded by the router. To prevent routing loops within an AS, IBGP peers do not advertise routes learned from the local AS.

With RR, IBGP peers can advertise routes learned from the local AS to each other. However, the Cluster_List attribute must be deployed to prevent routing loops within the AS.

An RR and its clients form a cluster. In an AS, each RR is uniquely identified by a Cluster_ID.

To prevent routing loops, the RR uses the Cluster_List attribute to record the Cluster_IDs of all RRs through which a route passes.

Similar to an AS_Path, which records all the ASs through which a route passes, a Cluster_List is composed of a series of Cluster_IDs and records all RRs through which a route passes. The Cluster_List is generated by the RR.

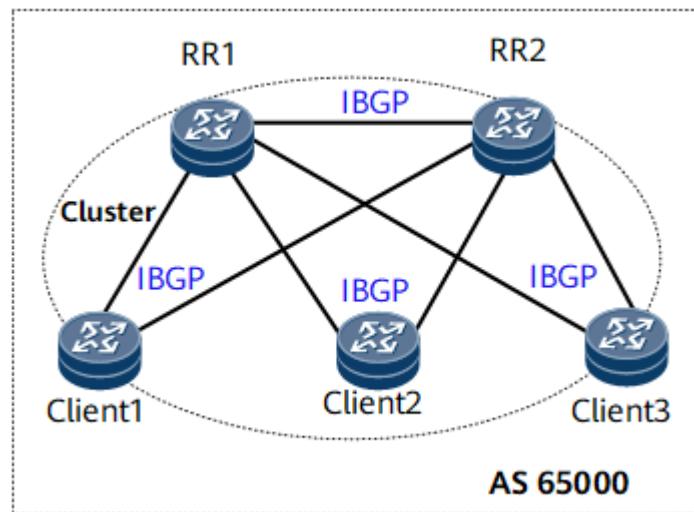
- Before an RR reflects a route between its clients or between its clients and non-clients, the RR adds the local Cluster_ID to the head of the Cluster_List. If a route does not carry any Cluster_List, the RR creates one for the route.
- After the RR receives an updated route, it checks the Cluster_List of the route. If the RR finds that its cluster ID is included in the Cluster_List, the RR discards the route. If its cluster ID is not included in the Cluster_List, the RR adds its cluster ID to the Cluster_List and then reflects the route.

Backup RR

To enhance network reliability and prevent single points of failure, more than one route reflector needs to be configured in a cluster. The route reflectors in the same cluster must be configured with the same Cluster_ID to prevent routing loops.

With backup RRs, clients can receive multiple routes to the same destination from different RRs. The clients then apply BGP route selection rules to choose the optimal route.

Figure 1-325 Backup RR



On the network shown in [Figure 1-325](#), RR1 and RR2 are in the same cluster. An IBGP connection is set up between RR1 and RR2. The two RRs are non-clients of each other.

- If Client 1 receives an updated route from an external peer, Client 1 advertises the route to RR1 and RR2 through IBGP.
- After receiving the updated route, RR1 adds the local Cluster_ID to the top of the Cluster_List of the route and then reflects the route to other clients (Client 1, Client 2, and Client 3) and the non-client (RR2).
- After receiving the reflected route, RR2 checks the Cluster_List and finds that its Cluster_ID is contained in the Cluster_List. In this case, it discards the updated route and does not reflect it to its clients.

If RR1 and RR2 are configured with different Cluster_IDs, each RR receives both the routes from its clients and the updated routes reflected by the other RR. Therefore, configuring the same Cluster_ID for RR1 and RR2 reduces the number of routes that each RR receives and memory consumption.

NOTE

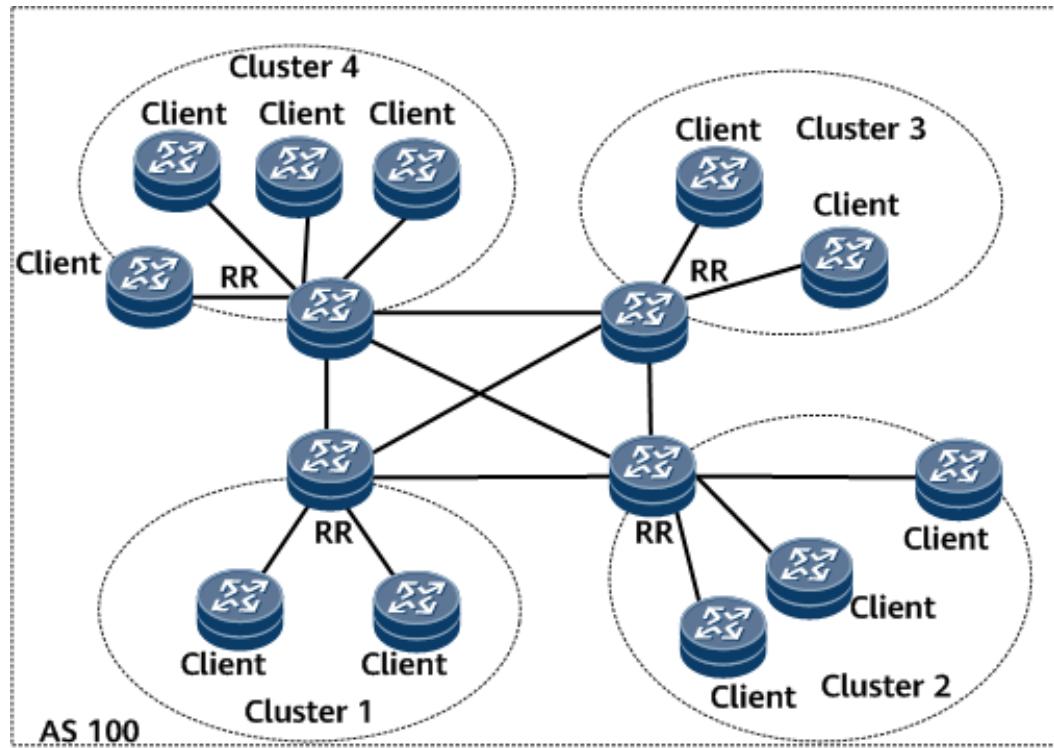
The application of Cluster_List prevents routing loops among RRs in the same AS.

Multiple Clusters in an AS

Multiple clusters may exist in an AS. RRs are IBGP peers of each other. An RR can be configured as a client or non-client of another RR. Therefore, the relationship between clusters in an AS can be configured flexibly.

For example, a backbone network is divided into multiple reflection clusters. Each RR has other RRs configured as its non-clients, and these RRs are fully meshed. Each client establishes IBGP connections only to the RR in the same cluster. In this manner, all BGP peers in the AS can receive reflected routes. [Figure 1-326](#) shows the networking.

Figure 1-326 Multiple clusters in an AS

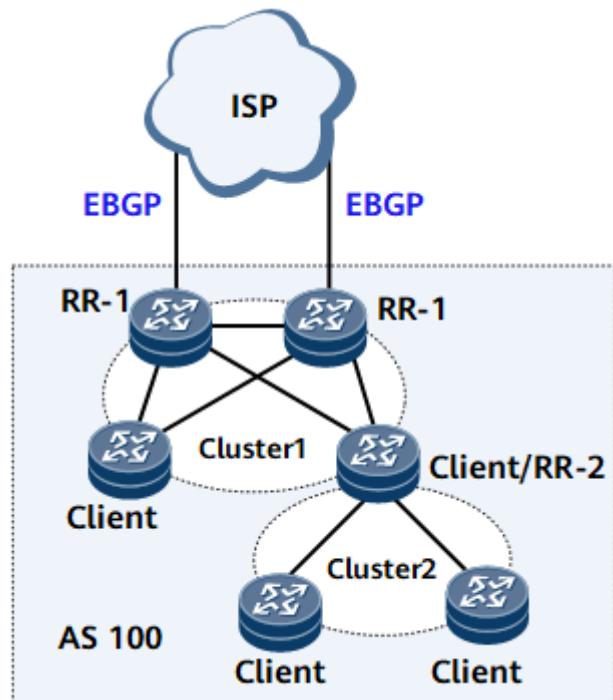


Hierarchical Reflector

Hierarchical reflectors are usually deployed if RRs need to be deployed. On the network shown in [Figure 1-327](#), the ISP provides Internet routes for AS 100. Two EBGP connections are established between the ISP and AS 100. AS 100 is divided into two clusters. The four routers in Cluster 1 are core routers.

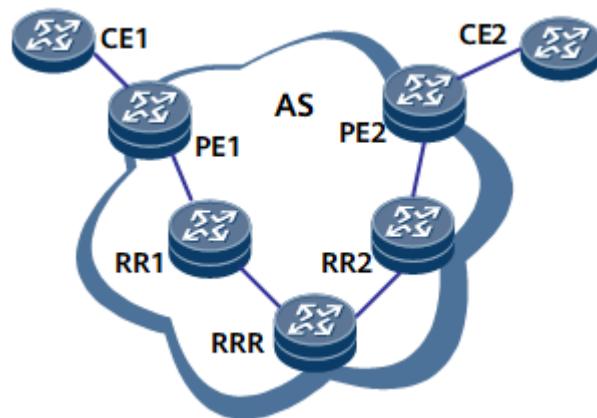
- Two Level-1 RRs (RR-1s) are deployed in Cluster 1, which ensures the reliability of the core layer of AS 100. The other two routers in the core layer are clients of RR-1s.
- One Level-2 RR (RR-2) is deployed in Cluster 2. RR-2 is a client of RR-1.

Figure 1-327 Hierarchical reflector



In [Figure 1-328](#), all PEs and RRs reside in the same AS, and peer relationships are established between each PE and its RR and between RRs in both VPNv4 and VPN-Target address families; PE1 is a client of the level-1 RR1, and PE2 is a client of the level-1 RR2; RRR is a level-2 RR, with RR1 and RR2 as its clients; RT 1:1 is configured on PE1 and PE2. PE1 receives a VPN route from CE1.

Figure 1-328 Networking with hierarchical RRs



If no RR cluster ID loop is allowed, after RR1 and RR2 advertise the RT routes learned from PEs to RRR (Level-2 RR), RRR implements route selection. If RRR selects the route learned from RR1, RRR advertises a VPN ORF route to RR1 and RR2. The Cluster_List of the route includes the local cluster ID. As a result, RR1 discards the VPN ORF route. Consequently, RR1 does not have the RT filter of RRR, unable to guide VPNv4 peers to advertise routes. As a result, CE2 fails to learn routes from CE1. To address this problem, run the **peer allow-cluster-loop**

command in the BGP-VPN-Target address family view on RR1. In the command, the peer address is set to the address of RRR. After the command is run, RR1 can accept the RT routes advertised by RRR (Level-2 RR) and can guide VPNv4 peers to advertise routes.

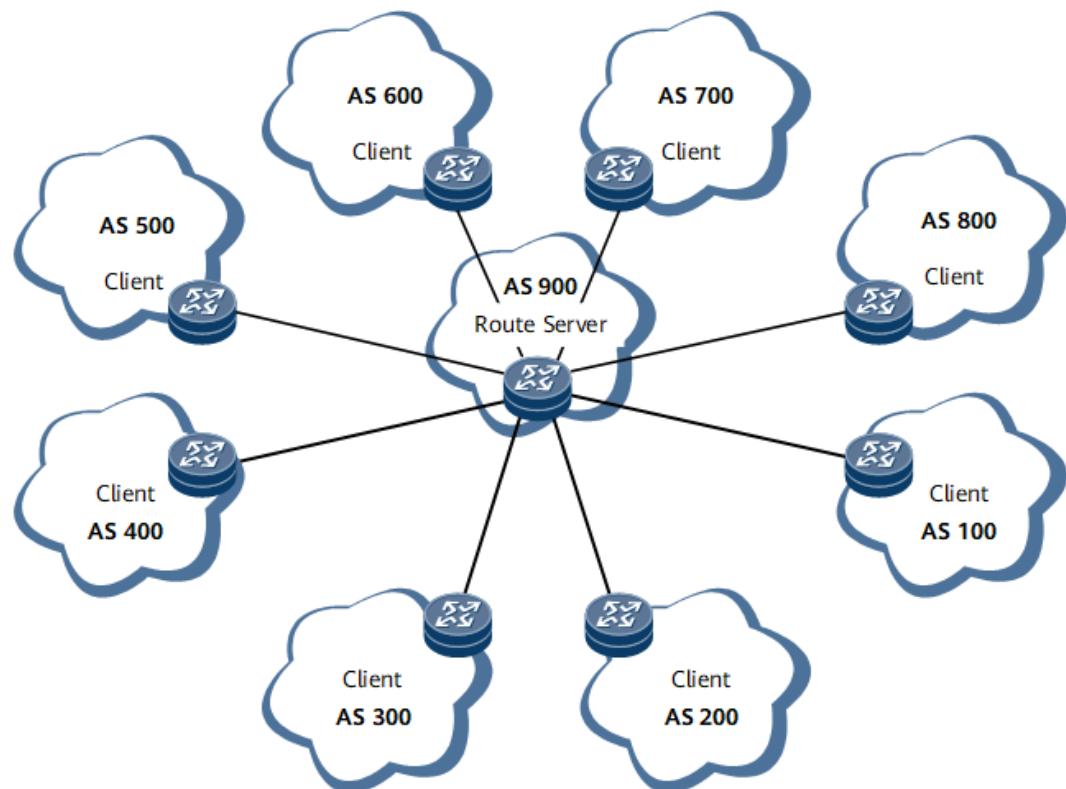
Route Server

The route server function is similar to the RR function used in IBGP full-mesh connection scenarios. It allows one or more routing devices to advertise routes to their clients (border devices) without changing path attributes, such as AS_Path, Nexthop, and MED, reducing the consumption of full-mesh connections on each border router.

Application

In some scenarios on the live network, to achieve network traffic interworking, EBGP full-mesh connections may be required. Full-mesh connections between border devices have high requirements on cost and device performance, and are not conducive to the expansion of the network topology and the number of devices. In [Figure 1-329](#), the route server can advertise routes to all its EBGP peers, without requiring EBGP full-mesh connections among ASBRs. Therefore, the route server function reduces network resource consumption.

Figure 1-329 Route server networking



BGP VPN Route Leaking

Route leaking refers to the process of adding a BGP VPN route to the routing table of the local or remote VPN instance in a BGP/MPLS IP VPN scenario. Route leaking can be classified as local route leaking or remote route leaking based on the source of the BGP VPN route.

- Remote route leaking: After a PE receives a BGP VPNV4/VPNV6 route from a remote PE, the local PE matches the export target (ERT) of the route against the import targets (IRTs) configured for local VPN instances. If the match succeeds, the device converts the BGP VPNV4/VPNV6 route to a BGP VPN route and adds the BGP VPN route to the routing table of the VPN instance.
- Local route leaking: A PE matches the ERT of a BGP VPN route in a local VPN instance against the IRTs configured for other local VPN instances. If the ERT matches the IRT of a local VPN instance, the PE adds the BGP VPN route to the routing table of this local VPN instance. Locally leaked routes include locally imported routes, summary routes, and routes learned from VPN peers.

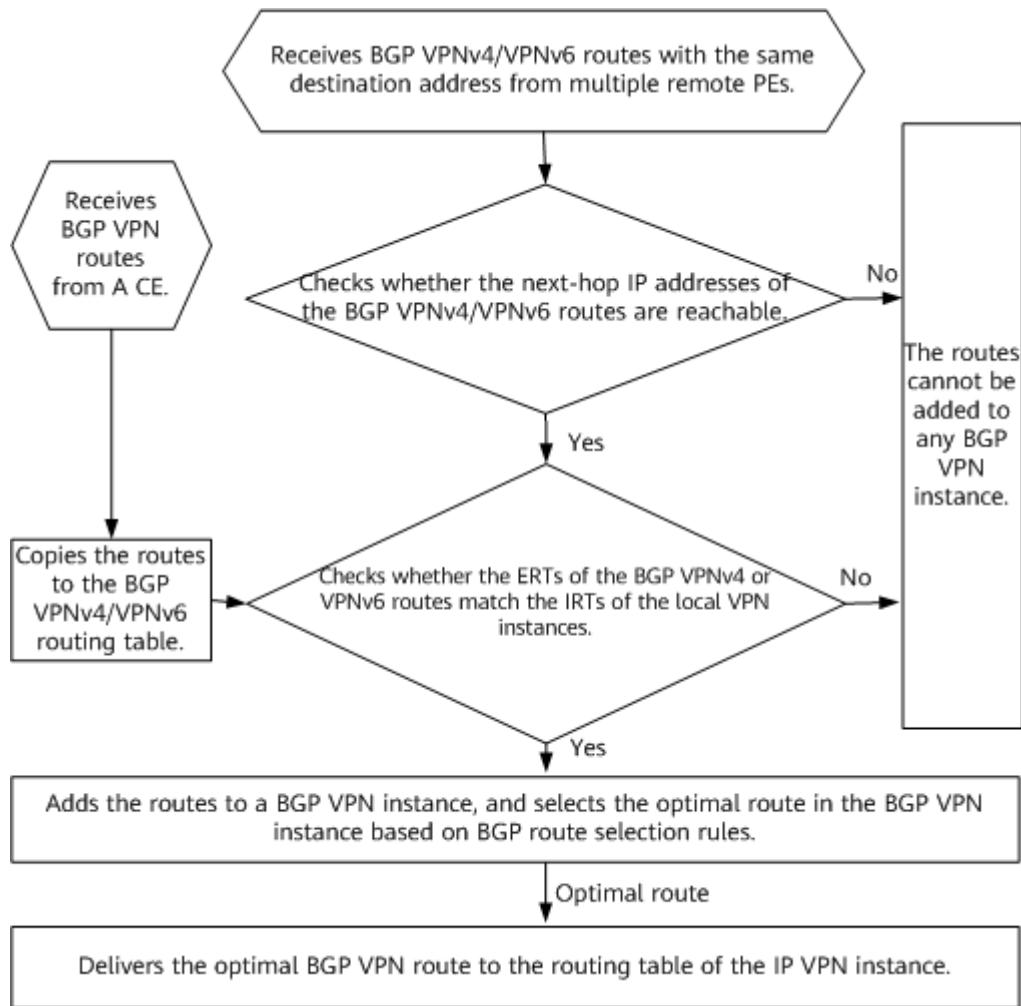
NOTE

If the routes to be imported from the IP routing table are load balancing routes, regardless of IGP routes or static routes, only the first route is imported.

Local route leaking means that imported routes are copied to the routing tables of other VPN instances.

After a PE receives VPNV4/VPNV6 routes destined for the same IP address from another PE or VPN routes from a CE, the local PE implements route leaking by following the steps shown in [Figure 1-330](#).

Figure 1-330 Flowchart for BGP VPN route leaking



In [Figure 1-331](#), PEs have the same VPN instance (**vpna**) and the RTs among VPN instances match each other. The RD configured for PE2 and PE3 is 2:2, and that configured for PE4 is 3:3. Site 2 has a route destined for 10.1.1.0/24. The route is sent to PE2, PE3, and PE4, which convert this route into a BGP VPNv4 route and send the VPNv4 route to PE1. On PE1, the route is leaked into the routing table of the BGP VPN instance on PE1, as shown in [Figure 1-332](#). The process is described as follows:

1. After receiving the BGP VPNv4 routes from PE2, PE3, and PE4, PE1 adds them to its BGP VPNv4 routing table.
2. PE1 converts the BGP VPNv4 routes to BGP VPN routes by removing their RDs, adds the BGP VPN routes to the routing table of the VPN instance, selects an optimal route from the BGP VPN routes based on BGP route selection rules, and adds the optimal BGP VPN route to the IP VPN instance routing table.

Figure 1-331 Route leaking networking

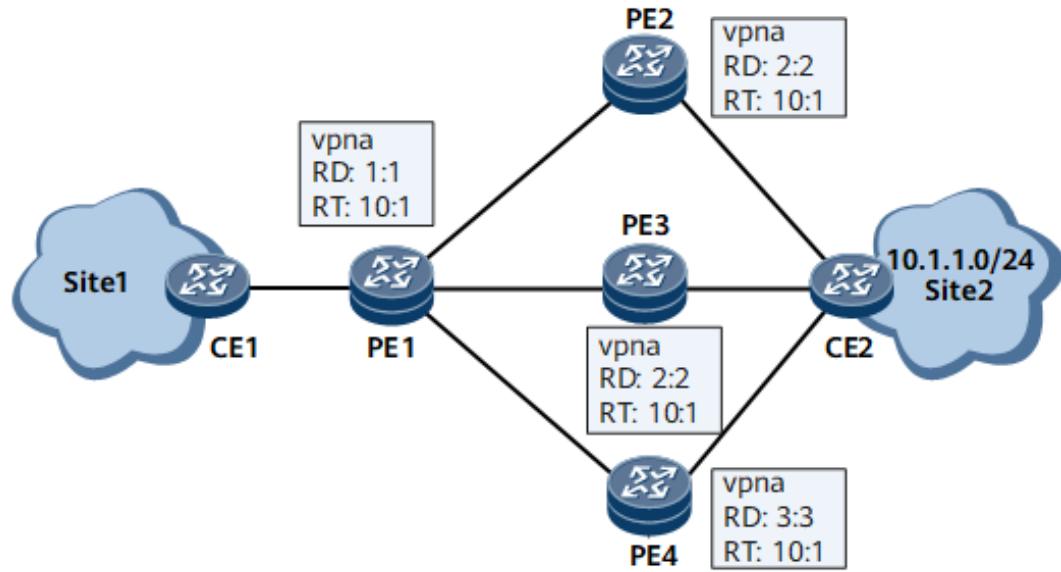
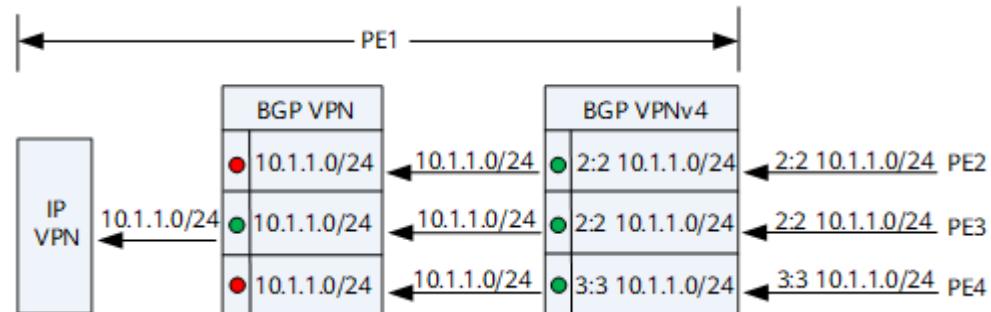


Figure 1-332 BGP VPN route leaking process



MP-BGP

Conventional BGP-4 manages only IPv4 unicast routing information, and inter-AS transmission of packets of other network layer protocols, such as multicast, is limited.

To support multiple network layer protocols, the Internet Engineering Task Force (IETF) extends BGP-4 to MP-BGP. MP-BGP is backward compatible. Specifically, routers supporting BGP extensions can communicate with the routers that do not support BGP extensions.

As an enhancement of BGP-4, MP-BGP provides routing information for various routing protocols, including IPv6 (BGP4+) and multicast.

- MP-BGP maintains both unicast and multicast routes. It stores them in different routing tables to separate unicast routing information from multicast routing information.

- MP-BGP supports both unicast and multicast address families and can build both the unicast routing topology and multicast routing topology.
- Most unicast routing policies and configuration methods supported by BGP-4 can be applied to multicast, and unicast and multicast routes can be maintained according to these routing policies.

Extended Attributes

BGP-4 Update packets carry three IPv4-related attributes: NLRI (Network Layer Reachable Information), Next_Hop, and Aggregator. Aggregator contains the IP address of the BGP speaker that performs route summarization.

To support multiple network layer protocols, BGP-4 needs to carry network layer protocol information in NLRI and Next_Hop. MP-BGP introduces the following two route attributes:

- MP_REACH_NLRI: indicates the multiprotocol reachable NLRI. It is used to advertise a reachable route and its next hop.
- MP_UNREACH_NLRI: indicates the multiprotocol unreachable NLRI. It is used to delete an unreachable route.

The preceding two attributes are optional non-transitive. Therefore, the BGP speakers that do not support MP-BGP will ignore the information carried in the two attributes and do not advertise the information to other peers.

Address Families

The Address Family Information field consists of a 2-byte Address Family Identifier (AFI) and a 1-byte Subsequent Address Family Identifier (SAFI).

BGP uses address families to distinguish different network layer protocols. For the values of address families, see relevant standards. The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports multiple MP-BGP extensions, such as VPN extension and IPv6 extension. The extensions are configured in their respective address family views.

NOTE

The supported address families are subject to the device.

1. For details about how to configure BGP in the IPv4 address family view, see "BGP Configuration." The BGP-IPv4 unicast address family has the following functions:
 - Maintains public network BGP peers and transmits public network IPv4 routing information. When the BGP-IPv4 unicast address family transmits public network IPv4 routes, the SAFI is 1.
 - Transmits public network labeled IPv4 routes. This function is mainly used in inter-AS BGP/MPLS IP VPN Option C or inter-AS BGP/MPLS IPv6 VPN Option C scenarios. When the BGP-IPv4 unicast address family transmits public network labeled IPv4 routes, the SAFI is 4.
2. For details about how to configure BGP in the IPv6 address family view, see "BGP4+ Configuration." The BGP-IPv6 unicast address family has the following functions:

- Maintains public network IPv6 BGP peers and transmits public network IPv6 routing information. When the BGP-IPv6 unicast address family transmits public network IPv6 routes, the SAFI is 1.
 - Transmits labeled IPv6 routes in 6PE scenarios. When the BGP-IPv6 unicast address family transmits labeled IPv6 routes, the SAFI is 4.
3. Multicast-related address family views, such as the BGP-IPv4 multicast address family view, BGP-MVPN address family view, BGP-IPv6 MVPN address family view, and BGP-MDT address family view, can transmit inter-AS routing information and are mainly used in MBGP, BIER, NG MVPN, BIERv6, and Rosen MVPN scenarios. For details about its application in multicast, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X seriesRouter Configuration Guide - IP Multicast*.
- The BGP-IPv4 multicast address family is used to configure multicast BGP (MBGP) peers. When a multicast source and receivers are located in different autonomous systems (ASs), an inter-AS multicast distribution tree needs to be established. MBGP can be used to transmit inter-AS routing information for multicast.
 - BGP-MVPN address family view: BGP A-D MVPN has two modes: MDT-SAFI A-D and MCAST-VPN SAFI A-D. In both BGP A-D MVPN modes, MVPN configurations, including RDs and Share-Group addresses, are transmitted between BGP peers to discover PE peer information. In this manner, MVPN services can run on public network tunnels based on PIM-SSM MDTs. Of the two modes, the MCAST-VPN SAFI A-D mode has a broader definition, supports more extensions, and carries more MVPN attributes and information for establishing public network tunnels. Therefore, the MCAST-VPN SAFI A-D mode can be used to support the next-generation MVPN.
 - The BGP-MDT address family is mainly used in PIM-SSM scenarios. It is used to transmit Share-Group source and group information on PEs through BGP sessions between the PEs. The following problems may occur in actual applications: To deploy PIM SSM on the public network, you must know the source address. The Share-Group is configured on a PE, but the PE does not know the multicast source addresses (addresses of MT interfaces) corresponding to the Share-Groups on other PEs. All PEs in a multicast domain (MD) select an interface address that is used to establish the public network BGP peer relationship as the multicast source IP address.
4. VPN-related address family views, such as the BGP-VPNV4 address family view, BGP-VPNV6 address family view, BGP-VPN instance view, BGP multi-instance VPN instance view, BGP-L2VPN-AD address family view, and BGP-L2VPN-AD address family view, are mainly used in BGP/MPLS IP VPN, VPWS, and VPLS scenarios. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X seriesRouter Feature Description - VPN*.
- The BGP-VPNV4 address family is mainly used to establish BGP VPNV4 peer relationships between PEs to exchange VPNV4 routes in BGP/MPLS IP VPN scenarios. In BGP VPNV4 extension, after a PE receives a CE's local VPN routes, the PE adds a route distinguisher (RD) and an export route target (ERT) to these routes and exchanges VPN routes with other PEs through BGP VPNV4 peer relationships. Routes of different VPNs are distinguished by RD on the public network, and import of routes from different CEs is controlled by RT. A PE maintains only the routing

- information about the VPNs that are directly connected to the PE but does not maintain all VPN routes on the service provider network.
 - The BGP-VPNv6 address family is mainly used in BGP/MPLS IPv6 VPN scenarios where PEs establish BGP VPNv6 peer relationships to exchange VPNv6 routes. In BGP VPNv6 extension, after a PE receives a CE's local IPv6 routes, the PE adds an RD and an RT to the routes and exchanges VPN routes with its BGP VPNv6 peers. Routes of different VPNs are distinguished by RD on the public network, and import of routes from different CEs is controlled by RT. A PE maintains only the routing information about the VPNs that are directly connected to the PE but does not maintain all VPN routes on the service provider network.
 - The BGP VPN instance IPv4 address family is mainly used in BGP/MPLS IP VPN scenarios where PEs and CEs use BGP to exchange VPN IPv4 routes.
 - The BGP VPN instance IPv6 address family is mainly used in BGP/MPLS IPv6 VPN scenarios where PEs and CEs use IPv6 BGP to exchange VPN IPv6 routes.
 - The BGP-L2VPN address family is used to manage L2VPN label blocks. MPLS/L2VPN can be implemented in multiple modes. BGP can be used as the exchange signaling. Similar to MPLS L3VPN, MPLS/L2VPN allows PEs to automatically discover L2VPN nodes and transmit VPN information through BGP sessions and use VPN targets to differentiate VPNs.
 - The BGP L2VPN-AD address family is mainly used to configure BGP AD VPLS. In this address family, information about BGP AD VPLS members is exchanged. BGP AD VPLS combines the advantages of multiple VPLS signaling modes. It uses extended BGP messages to discover VSI members and uses LDP FEC 129 to negotiate PW establishment to implement automatic VPLS PW deployment.
 - VPLS is a point-to-multipoint L2VPN service provided on a public network. The BGP-VPLS address family is mainly used in VPLS scenarios. After BGP peer relationships are established in the BGP-VPLS address family view of PEs, BGP is used to exchange VPLS label block information.
 - The BGP-VPN-Target address family is mainly used to configure VPN ORF. VPN ORF enables a PE to receive only desired routes, which reduces the pressure on the routing tables of the PE, the intra-AS RR, and inter-AS ASBR.
5. EVPN-related address families, such as the BGP-EVPN address family view and BGP multi-instance EVPN address family view, are mainly used to configure BGP EVPN peers and apply to EVPN VPLS, EVPN VPWS, and EVPN L3VPN. EVPN is a VPN technology used for Layer 2 network interworking. EVPN is similar to BGP/MPLS IP VPN. EVPN defines a new type of BGP network layer reachability information (NLRI), called the EVPN NLRI. The EVPN NLRI defines new types of BGP EVPN routes to implement MAC address learning and advertisement between Layer 2 networks at different sites. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X seriesRouter Feature Description - VPN - EVPN Feature Description*.
6. The BGP IPv4 SR Policy address family view and BGP IPv6 SR Policy address family view are mainly used in Segment Routing MPLS and Segment Routing IPv6 scenarios. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X seriesRouter Feature Description - Segment Routing*.

7. Flow-related address family views, such as the BGP-Flow address family view, BGP-Flow VPNv4 address family view, BGP-Flow VPNv6 address family view, BGP-Flow VPN instance IPv4 address family view, and BGP-Flow VPN instance IPv6 address family view are mainly used to defend against DoS/DDoS attacks and improve network security and availability. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description - Security - BGP Flow Specification Feature Description*.
8. The BGP-labeled address family view and BGP-labeled-VPN instance IPv4 address family view are mainly used for carrier configuration using the BGP label distribution solution. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration - VPN - BGP/MPLS IP VPN Configuration* and *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration - VPN - EVPN Configuration*.
9. The BGP-LS address family view is mainly used to summarize the topology information collected by an IGP and send the information to the upper-layer controller. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description - BGP Feature Description - BGP-LS*.

BGP Security

Route leaking and route hijacking may have serious impacts on ISP operations. They can redirect traffic and may cause denial of service (DoS), data leakage, increased latency, and data loss. Applying BGP security can mitigate the problems caused by attackers.

BGP Authentication

BGP can work properly only after BGP peer relationships are established. Authenticating BGP peers can improve BGP security. BGP supports the following authentication modes:

- MD5 authentication

BGP uses TCP as the transport layer protocol. Message Digest 5 (MD5) authentication can be used when establishing TCP connections to improve BGP security. MD5 authentication sets the MD5 authentication password for the TCP connection, and TCP performs the authentication. If the authentication fails, the TCP connection cannot be established.

NOTE

The encryption algorithm used for MD5 authentication poses security risks. Therefore, you are advised to use an authentication mode based on a more secure encryption algorithm.

- Keychain authentication

Keychain authentication is performed at the application layer. It prevents service interruptions and improves security by periodically changing the password and encryption algorithms. When keychain authentication is configured for BGP peer relationships over TCP connections, BGP messages as well as the process of establishing TCP connections can be authenticated. For details about keychain, see "Keychain" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - Security*.

- **TCP-AO authentication**

The TCP authentication option (TCP-AO) is used to authenticate received and to-be sent packets during TCP session establishment and data exchange. It supports packet integrity check to prevent TCP replay attacks. TCP-AO authentication improves the security of the TCP connection between BGP peers and is applicable to the network that requires high security.

BGP GTSM

During network attacks, attackers may simulate BGP messages and continuously send them to the router. If the messages are destined for the router, it directly forwards them to the control plane for processing without validating them. As a result, the increased processing workload on the router's control plane results in high CPU usage. The Generalized TTL Security Mechanism (GTSM) defends against attacks by checking the time to live (TTL) value in each packet header. TTL refers to the maximum number of routers through which a packet can pass.

GTSM checks whether the TTL value in each IP packet header is within a pre-defined range, which protects services above the IP layer and improves system security.

After a GTSM policy of BGP is configured, an interface board checks the TTL values of all BGP messages. According to actual networking requirements, you can set the default action (to drop or pass) that GTSM will take on the messages whose TTL values are not within a pre-defined range. If a valid TTL range is specified based on the network topology and the default action that GTSM will take is set to drop, the BGP messages whose TTL values are not within the valid range are discarded directly by the interface board upon receipt. This prevents bogus BGP messages from consuming CPU resources.

You can enable the logging function so that the device can record information about message dropping in logs. The recorded logs facilitate fault locating.

BGP RPKI

Resource Public Key Infrastructure (RPKI) ensures BGP security by verifying the validity of the BGP route source AS, AS_Path, or route advertiser.

Attackers can steal user data by advertising routes that are more specific than those advertised by carriers. RPKI can resolve this problem. For example, a carrier has advertised a route destined for 10.10.0.0/16, and an attacker advertises a route destined for 10.10.153.0/24, which is more specific than 10.10.0.0/16. According to the longest match rule, the route 10.10.153.0/24 is preferentially selected for traffic forwarding. As a result, the attacker succeeds in illegally obtaining user data.

To solve the preceding problem, you can configure Route Origin Authorization (ROA), Autonomous System Provider Authorization (ASPA), or regional validation. This helps ensure BGP security.

ROA validation

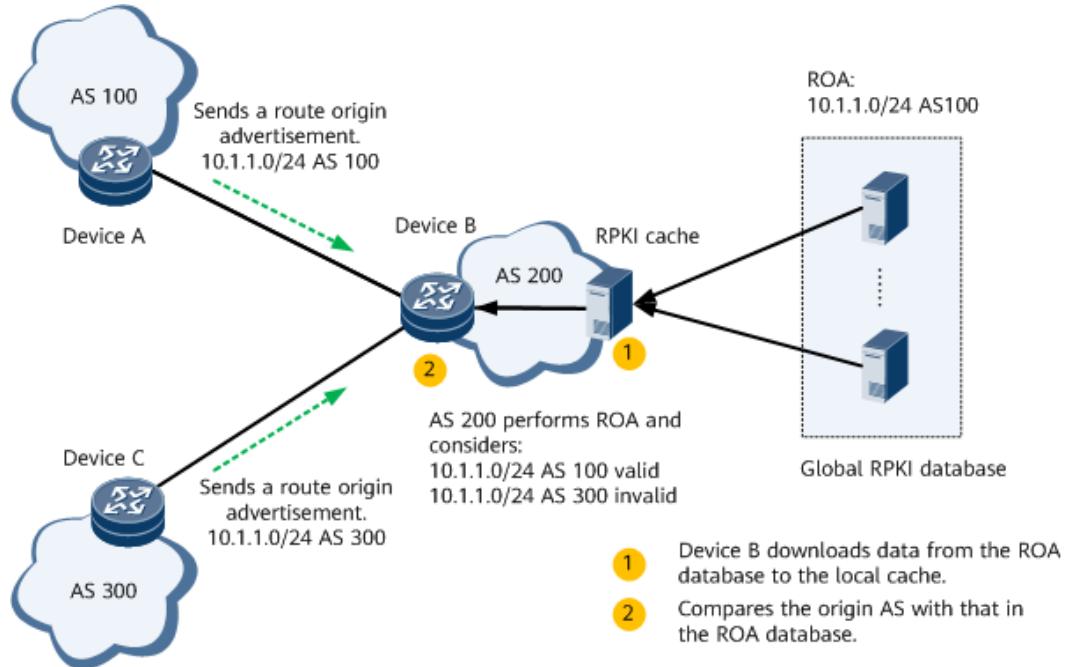
ROA stores the mapping between prefix addresses and the origin AS and checks whether the routes with a specified IP address prefix are valid by verifying the AS number.

In [Figure 1-333](#), a connection is created between Device B and the RPKI server. Device B can download ROA data from the RPKI database and verify the mapping between 10.1.1.0/24 and AS 100. When AS 200 receives a route with the prefix 10.1.1.0/24 from AS 100 and AS 300, it compares the origin AS with that in the ROA database. If they are the same, the route is considered valid. If they are different, the route is considered invalid. The route to 10.1.1.0/24 learned from AS 100 is valid because it matches the record in the ROA database, and the route advertisement from the origin AS 100 is considered valid. The route to 10.1.1.0/24 learned from AS 300 is invalid because it does not match the record in the ROA database, and the route advertisement from the origin AS 300 is considered invalid.

 NOTE

If no RPKI server is available, a static ROA database can be configured for Device B. In this way, ROA validation can be implemented based on the static ROA database, without relying on an RPKI server, thereby preventing route hijacking to a certain extent.

Figure 1-333 ROA validation



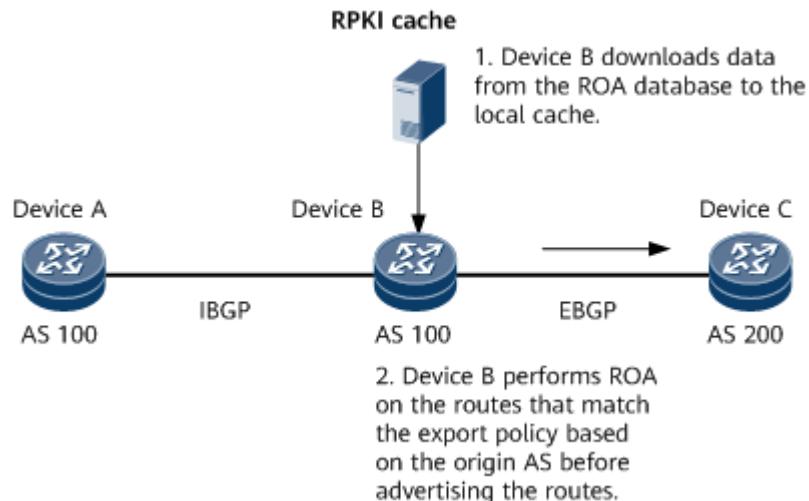
The ROA validation results for received routes are as follows:

- Valid: indicates that the route advertisement from the origin AS to the specified IP address prefix is valid.
- Invalid: indicates that the route advertisement from the origin AS to the specified IP address prefix is invalid and that the route is not allowed to participate in route selection.
- Not Found: indicates that the origin AS does not exist in the ROA database and that the route participates in route selection.

In addition, ROA validation can be applied to the routes to be advertised to an EBGP peer, which prevents route hijacking. In [Figure 1-334](#), Device B is configured to perform ROA validation on the routes to be advertised. Before advertising a

route that matches the export policy to an EBGP peer, Device B performs ROA validation on the route by matching the origin AS of the route against that of the corresponding route in the ROA database. If the route is not found in the ROA database, the validation result is Not Found. If the route is found in the ROA database and the origin AS of the route is the same as that of the corresponding route in the ROA database, the validation result is Valid. If the origin AS of the route is different from that of the corresponding route in the ROA database, the validation result is Invalid. If the validation result is Valid, the route is advertised by default. If the validation result is Not Found, the route is not advertised by default. If the validation result is Invalid, the route is not advertised by default and an alarm is generated; the alarm is cleared when all routes with the validation result of Invalid are withdrawn.

Figure 1-334 Outbound ROA validation



ASPA validation

RPKI-based ROA validation can detect unexpected path leaks, but it relies on the origin AS in the BGP attribute — AS_Path, which may be manipulated by attackers to forge the origin or path segment. As an inter-AS routing security mechanism, ROA provides only origin validation but not route path validation. A shared signature database that contains objects of customer-to-provider relationships is constructed through RPKIv2. Through the database, ASPA can detect invalid AS_Paths in routes received from peers, detect unexpected route leaking, and provide route path validation.

An ASPA record is a digitally signed object that binds a set of provider AS numbers to a customer AS number and is signed by the holder of the customer AS. ASPA uses an ASPA pair (customer AS, provider AS) to prove that the customer AS holder has authorized a specific provider AS to advertise the customer AS's routes onward. ASPA can automatically detect invalid AS_Paths in the routes received from customers and providers.

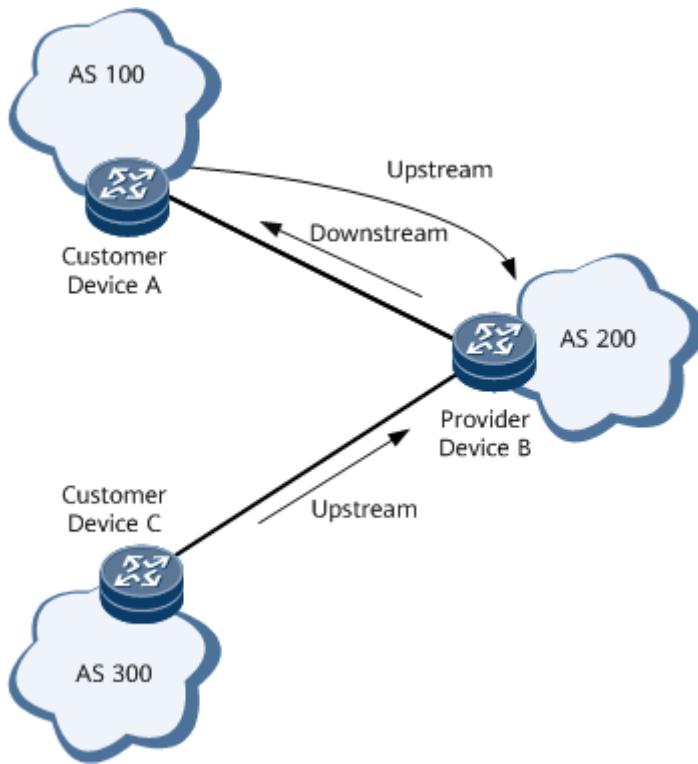
During ASPA validation, the BGP roles that an AS can have in relation to another AS are defined. The local AS refers to the AS where the local device resides, and the remote AS refers to the AS where the peer resides. The roles are described as follows:

- Provider: The local AS is the transit provider of the remote AS and can advertise any available routes to a customer.
- Route server (RS): The local AS is a route server and can advertise any available routes to an RS-client (remote AS).
- RS-client: The local AS is an RS-client and can advertise any routes learned from a customer or locally originated routes to an RS.
- Customer: The local AS is a transit customer of the remote AS and can advertise any routes learned from a customer or locally originated routes to a provider.
- Lateral-peer: If the local AS and the remote AS have a lateral peer relationship, any routes learned from a customer or locally originated routes can be advertised to the peer end.
- Sibling: If the local and remote ASs have the sibling peer relationship, they can advertise all routes (both customer and non-customer routes) to each other.

After a device establishes a connection with an RPKI server, the device obtains ASPA data from the RPKI server, and saves the data locally. After receiving a route from a peer, the device parses the AS_Path in the route and verifies the validity of the AS_Path. The detailed process is as follows:

1. Each AS on the network registers ASPA pairs and uploads them to the RPKI server. Each ASPA pair is in the format of (customer AS, provider AS). On the network shown in the figure, Device B has customer-to-provider relationships with Device A and Device C. Therefore, (100, 200) and (300, 200) are registered in the RPKI database. In the notation {AS(1), AS(2), ..., AS(N)}, if each AS (i + 1) is a provider of AS (i), the corresponding path is referred to as an upstream path; if each AS (i - 1) is a provider of AS (i), the corresponding path is referred to as a downstream path. On the network shown in the figure, the AS_Path is (300, 200, 100), and AS 200 is a provider of AS 300. Therefore, the path corresponding to (300, 200) is an upstream path. AS 200 is a provider of AS 100. Therefore, the path corresponding to (200, 100) is a downstream path. Similarly, the path corresponding to (100, 200) is an upstream path.

Figure 1-335 Registering ASPA pairs



2. After receiving a route from a peer, a device parses the AS_Path. For example, if the AS_Path is (a₁, a₂, a₃, ..., a_n), the device validates the route based on the peer role. If the route is received from a customer, lateral peer, RS-client, or RS, the device uses the upstream path algorithm for validation. If the route is received from a provider or sibling, the device uses the downstream path algorithm for validation. On the network shown in the figure, Device A receives a route from a provider and uses the downstream path algorithm to verify the validity of the AS_Path (200, 300) based on the ASPA data in the RPKI database.

On the network shown in [Figure 1-336](#), Device B establishes a connection with the RPKI server and obtains ASPA data {(600, 100), (400, 600), (500, 400), (400, 700)} from the RPKI server. When Device B in AS 200 receives routes with prefix 10.1.1.0/24 from AS 100 and AS 300, Device B checks whether the AS_Path of each route is consistent with the ASPA data in the ASPA database to verify the validity of the AS_Path. If the AS_Path of a route is considered valid, the ASPA validation result is Valid (indicating that the route is valid). If the AS_Path of a route is considered invalid, the ASPA validation result is Invalid (indicating that the route is invalid). The detailed process is as follows:

Device A and Device B have a customer-to-provider relationship, and the path from Device A to Device B is an upstream path. After receiving the route from Device A, Device B parses the AS_Path (100, 600, 400). According to (400, 600) and (600, 100) in the ASPA data, Device A considers the route received from Device A valid.

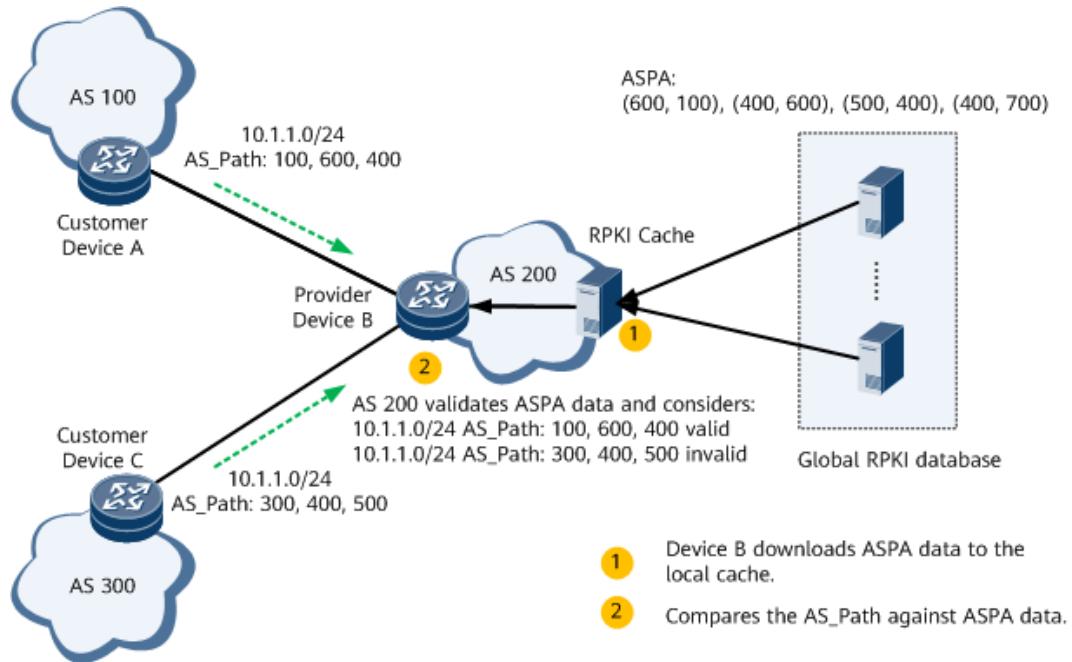
Device C and Device B have a customer-to-provider relationship, and the path from Device C to Device B is an upstream path. After receiving the route from Device C, Device B parses the AS_Path (300, 400, 500). (500, 400) exists in the

ASPA data, but (400, 300) does not. Therefore, the AS_Path of the route received from Device C is invalid, and this route is also invalid.

 NOTE

If no RPKI server is available, a static ASPA database can be configured for Device B. A static ASPA database helps implement ASPA validation and intercept invalid routes, without requiring the RPKI server.

Figure 1-336 ASPA validation



The ASPA validation results for received routes are as follows:

- Valid: The AS_Path attribute carried in the route is valid.
- Invalid: The AS_Path attribute carried in the route is invalid, and the route cannot participate in route selection.
- Not Found: The AS_Path carried in the route cannot be verified based on the obtained ASPA data, but the route participates in route selection.

Regional validation

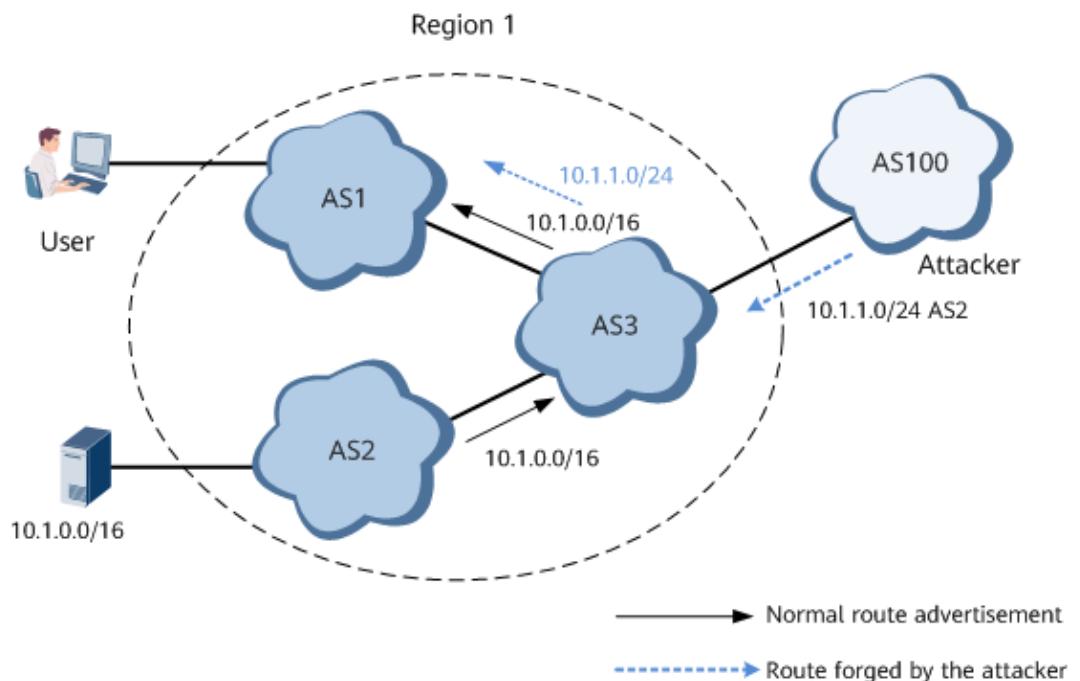
Regional validation: Users can manually configure regions by combining multiple trusted ASs into a region and combining multiple regions into a regional confederation. Regional validation controls route selection results by checking whether the routes received from EBGP peers in an external region belong to the local region. This prevents intra-region routes from being hijacked by attackers outside the local region, and ensures that hosts in the local region can securely access internal services.

Regional validation applies to the following typical scenarios: regional validation scenario and regional confederation validation scenario.

- As shown in [Figure 1-337](#), AS 1, AS 2, and AS 3 belong to a carrier's network, and AS 3 connects to AS 100 of another carrier's network. The user accesses the server at 10.1.0.0/16 in AS 2 through AS 1. In normal cases, the user

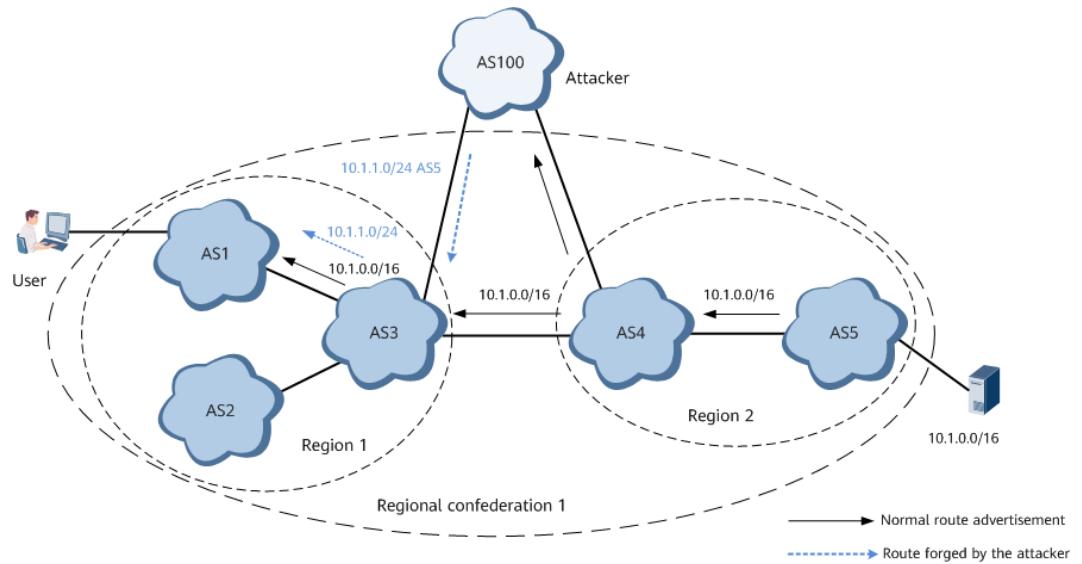
accesses the server through the path AS 1 -> AS 3 -> AS 2. If an attacker forges a route 10.1.1.0/24 in AS 100, the user-to-server traffic will be illegally obtained by the attacker because the route advertised by the attacker is more specific. To solve this problem, configure regional validation on the border device of AS 3 to combine AS 1, AS 2, and AS 3 into region 1. AS 3 receives the attack route from AS 100, and the route source is AS 2, indicating that the route belongs to the local region. However, as the route is received from the BGP peer outside region 1, the route is considered invalid through regional validation. As a result, the route is set to be invalid, or the priority of the route is reduced.

Figure 1-337 Regional validation scenario



- As shown in **Figure 1-338**, AS 1, AS 2, and AS 3 belong to a carrier's network, and AS 4 and AS 5 belong to a partner carrier's network. Both AS 3 and AS 4 are connected to AS 100 of another carrier. The user accesses the server at 10.1.0.0/16 in AS 5 through AS 1. In normal cases, the user accesses the server through the path AS 1 -> AS 3 -> AS 4 -> AS 5. If an attacker forges a route 10.1.1.0/24 in AS 100, the user-to-server traffic will be illegally obtained by the attacker because the route advertised by the attacker is more specific. To solve this problem, configure regional validation on the border device of AS 3 to combine AS 1, AS 2, and AS 3 into region 1, and AS 4 and AS 5 into region 2, and then group the regions 1 and 2 into regional confederation 1. AS 3 receives the attack route from AS 100, and the route source is AS 5, indicating that the route belongs to the local regional confederation. However, as the route is received from the BGP peer outside the regional confederation, the route is considered invalid by regional validation. As a result, the route is set to be invalid, or the priority of the route is reduced.

Figure 1-338 Regional confederation validation scenario



SSL/TLS Authentication

Secure Sockets Layer (SSL) is a security protocol that protects data privacy on the Internet. Transport Layer Security (TLS) is a successor of SSL. TLS protects data integrity and privacy by preventing attackers from eavesdropping the data exchanged between a client and server. To ensure data transmission security on a network, SSL/TLS authentication can be enabled for BGP message encryption.

BGP GR

Graceful restart (GR) is one of the high availability (HA) technologies, which comprise a series of comprehensive technologies such as fault-tolerant redundancy, link protection, faulty node recovery, and traffic engineering. GR is a fault-tolerant redundancy technology. It ensures uninterrupted transmission of key services during the restart of a routing protocol. Currently, GR is widely used in active/standby switchover and system upgrade scenarios.

GR is usually used when the active route processor (RP) fails because of a software or hardware error, or used by an administrator to perform the active/standby switchover.

NOTE

You are not advised to configure both BFD and BGP GR on the same device. If both are configured and a BFD session detects a fault on an interface, the session exits GR. As a result, traffic forwarded based on GR is interrupted, which may cause network problems.

Prerequisite for Implementation

On a traditional device, a processor implements both control and forwarding. The processor finds routes based on routing protocols, and maintains the routing table and forwarding table of the device. Mid-range and high-end devices generally adopt the multi-RP structure to improve forwarding performance and reliability. The processor in charge of routing protocols is located on the main control board, whereas the processor responsible for data forwarding is located on the interface.

board. The design helps to ensure the continuity of packet forwarding on the interface board during the restart of the main processor. The technology that separates control from forwarding satisfies the prerequisite for GR implementation.

Currently, a GR-capable device must have two main control boards. In addition, the interface board must have an independent processor and memory.

Related Concepts

The concepts related to GR are as follows:

- GR Restarter: indicates a device that performs an active/standby switchover triggered by the administrator or a failure. A GR Restarter must support GR.
 - Force-Quit timer: indicates the timer used to exit GR forcibly in a scenario where the device cannot exit GR automatically due to internal errors.
 - Protection timer: indicates the timer used for protection in a scenario where no peers are up.
 - Wait-All-Peer-Up timer: indicates the timer used for waiting for all peers to go up in a scenario where some peers are not up.
 - Selection-Deferral timer: indicates the timer used for protection in a scenario where a Helper does not send End-of-RIB (EOR) messages.
 - EOR Timer: indicates the maximum period during which a GR Restarter waits for EOR messages from a GR Helper. If a GR Restarter does not receive any EOR message from a GR Helper within the EOR timer, the GR Restarter selects routes based on the existing routes.
- GR Helper: indicates a peer of a GR Restarter. A GR Helper must support GR.
 - GR timer: indicates the period during which a GR Helper retains the topology information or routes obtained from a GR Restarter after the GR Helper finds that the GR Restarter is down.
 - EOR timer: indicates the maximum period during which a GR Helper waits for EOR messages from a GR Restarter. If a GR Helper does not receive any EOR message from a GR Restarter within the EOR timer, the GR Helper selects routes based on the existing routes.
- GR session: indicates a session, through which a GR Restarter and a GR Helper negotiate GR capabilities.
- EOR: indicates a type of BGP message used to notify a peer that the first route upgrade after BGP session establishment is complete.

Fundamentals

Fundamentals of a BGP GR Helper are as follows:

1. During BGP peer relationship establishment, devices negotiate GR capabilities by sending supported GR capabilities to each other.
2. When detecting that the GR Restarter is down, the GR Helper starts the GR timer and waits for the Restarter to go up.
 - Before the GR timer expires, the device retains the routes and forwarding entries related to the GR Restarter until the GR Restarter goes up. The process then goes to Step 3.

- When the GR timer expires, the device deletes the routes and forwarding entries related to the GR Restarter and exits the GR Helper state.
3. After the GR Restarter goes up, the GR Helper receives routes from the GR Restarter and starts the EOR timer. The GR Helper exits the GR Helper state and ages the routes related to the GR Restarter when one of the following conditions is met:
- The GR Helper receives an EOR message from the GR Restarter, and the EOR timer is deleted.
 - The EOR timer expires but the GR Helper fails to receive an EOR message from the GR Restarter.

Fundamentals of a BGP GR Restarter are as follows:

1. When a BGP process is restarted, a GR-capable device enters the GR Restarter state, starts the Protection timer and Force-Quit timer, and waits for BGP peer relationships to be reestablished.
 - After the first BGP peer relationship is established, the Protection timer is deleted, and the process goes to Step 2.
 - No BGP peer relationship is established, and the Protection timer expires. In this case, the GR Restarter state exits.
2. The GR Restarter continues to wait for the remaining BGP peer relationships to be established, receives routes and EOR messages from the BGP peers with which BGP peer relationships have been established, starts the Wait-All-Peer-Up Timer and Selection-Deferral Timer, and waits for the establishment of all BGP peer relationships and EOR messages. The GR Restarter starts route selection when one of the following conditions is met:
 - The BGP peer relationships are established, and the GR Restarter receives EOR messages from the peers.
 - The Wait-All-Peer-Up timer expires.
 - The Selection-Deferral timer expires.
3. After the GR restarter completes route selection, it re-advertises routes to its BGP peers and exits the GR restarter state.

GR Reset

Currently, BGP does not support dynamic capability negotiation. Therefore, each time a capability in an address family changes (enabled or disabled for example) on a BGP speaker, the BGP speaker terminates the session with the corresponding peer and renegotiates the capability with the peer.

In some scenarios, BGP IPv4 unicast peer sessions have been established and IPv4 services are running properly. If a BGP peer relationship in another address family is established based on this session, the BGP IPv4 unicast peer relationship will be re-established, affecting IPv4 services. To solve the preceding problem, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X provides the function of resetting a BGP connection in GR mode.

With the function of resetting a BGP connection in GR mode, if a BGP peer relationship in another address family is established based on an IPv4 BGP unicast peer session, BGP enters the GR state and renegotiates BGP capabilities with its peer. Throughout the entire process, BGP re-establishes the BGP IPv4 unicast peer session but retains the original routing entries. The forwarding module forwards

packets based on the routing entries, thereby ensuring that IPv4 services are not interrupted.

Benefits

Through BGP GR, the forwarding is not interrupted. In addition, the flapping of BGP occurs only on the peers of the GR Restarter, and does not occur in the entire routing domain. This is important for BGP that needs to process a large number of routes.

BFD for BGP

BGP periodically sends Keepalive messages to a peer to monitor its status. This mechanism, however, takes more than 1 second to detect a fault. If data is transmitted at Gbit/s rates, such a lengthy detection period will result in a large amount of data being lost, making it impossible to meet the high reliability requirements of carrier-grade networks.

BFD can detect faults within milliseconds. After BFD for BGP is enabled, BFD can quickly detect faults on links between BGP peers and report the faults to BGP, implementing fast BGP route convergence.

NOTE

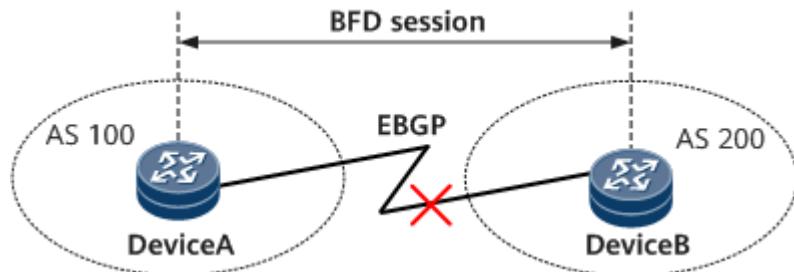
You are not advised to configure both BFD and BGP GR on the same device. If both are configured and a BFD session detects a fault on an interface, the session exits GR. As a result, traffic forwarded based on GR is interrupted, which may cause network problems.

Fundamentals

On the network shown in [Figure 1-339](#), DeviceA and DeviceB belong to AS 100 and AS 200, respectively. An EBGP connection is established between the two routers.

BFD is used to monitor the BGP peer relationship between DeviceA and DeviceB. If the link between them becomes faulty, BFD can quickly detect the fault and notify it to BGP.

Figure 1-339 Networking diagram of BFD for BGP



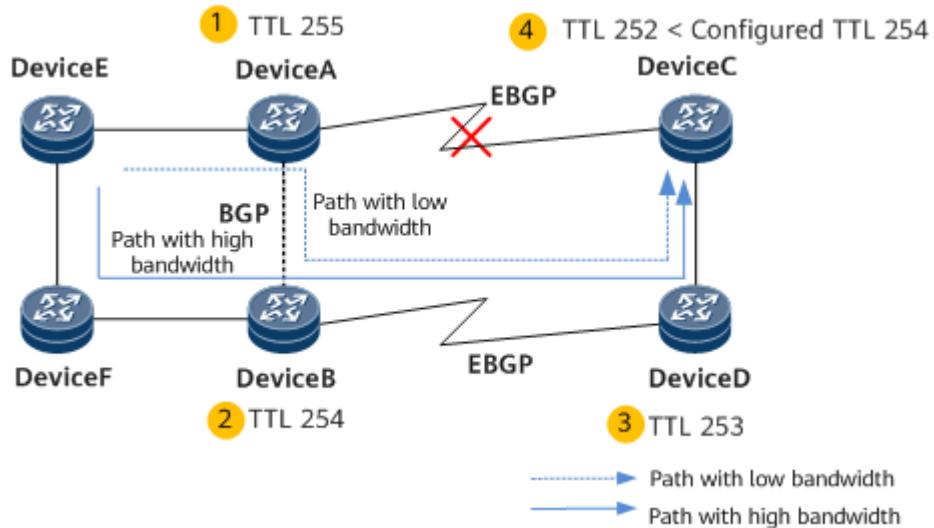
On the network shown in [Figure 1-340](#), indirect multi-hop EBGP connections are established between DeviceA and DeviceC, and between DeviceB and DeviceD; a BFD session is established between DeviceA and DeviceC; a BGP peer relationship is established between DeviceA and DeviceB; the bandwidth between DeviceA and

DeviceB is low. If the original forwarding path DeviceA->DeviceC fails, traffic is switched to the path DeviceE -> DeviceA-> DeviceB -> DeviceD -> DeviceC. Due to the low bandwidth on the link between DeviceA and DeviceB, traffic loss may occur.

To prevent this issue, you can set a TTL value on DeviceC for checking the BFD session with DeviceA. If the number of forwarding hops of a BFD packet (TTL value in the packet) is smaller than the TTL value set on DeviceC, the BFD packet is discarded, and BFD detects a session down event and notifies BGP. DeviceA then sends BGP Update messages to DeviceE for route update so that the traffic forwarding path is changed to DeviceE -> DeviceF -> DeviceB -> DeviceD -> DeviceC.

For example, the TTL value for checking the BFD session on DeviceC is set to 254. If the link between DeviceA and DeviceC fails, traffic sent from DeviceE is forwarded through the path DeviceE -> DeviceB -> DeviceD -> DeviceC. In this case, the TTL value in a packet decreases to 252 when the packet reaches DeviceC. Since 252 is smaller than the configured TTL value 254, the BFD packet is discarded, and BFD detects a session down event and notifies BGP. DeviceA then sends BGP Update messages to DeviceE for route update so that the traffic forwarding path is changed to DeviceE -> DeviceF -> DeviceB -> DeviceD -> DeviceC.

Figure 1-340 Networking diagram of setting a TTL value for checking the BFD session with a BGP peer



NOTE

BFD for BGP TTL check applies only to the scenario in which DeviceA and DeviceC are indirectly connected EBGP peers.

BGP Peer Tracking

BGP peer tracking provides quick detection of link or peer faults for BGP to speed up network convergence. If BGP peer tracking is enabled on a local device and a fault occurs on the link between the device and its peer, BGP peer tracking can quickly detect that routes to the peer are unreachable and notify BGP of the fault, thereby achieving rapid convergence.

Compared with BFD, BGP peer tracking is easy to configure because it needs to be configured only on the local device rather than on the entire network. Although both are fault detection mechanisms, BGP peer tracking is implemented at the network layer, whereas BFD is implemented at the link layer. For this reason, BGP peer tracking provides a slower convergence performance than BFD, making it inapplicable to services that require a rapid convergence, such as voice services.

Networking

BGP peer tracking can quickly detect link or peer faults by checking whether routes to peers exist in the IP routing table. If no route is found in the IP routing table based on the IP address of a BGP peer (or a route exists but is unreachable, for example, the outbound interface is a Null0 interface), the BGP session goes down, achieving fast BGP route convergence. If a reachable route can be found in this case, the BGP session does not go down.

On the network shown in [Figure 1-341](#), IGP connections are established between DeviceA, DeviceB, and DeviceC, a BGP peer relationship is established between DeviceA and DeviceC, and BGP peer tracking is configured on DeviceA. If the link between DeviceA and DeviceB fails, the IGP performs fast convergence first. As no route is found on DeviceA based on the IP address of DeviceC, BGP peer tracking detects that no reachable route to DeviceC is available and then notifies BGP on DeviceA of the fault. As a result, DeviceA terminates the BGP connection with DeviceC.

Figure 1-341 Network diagram of BGP peer tracking



NOTE

- If a default route exists on DeviceA and the link between DeviceA and DeviceB fails, BGP peer tracking will not terminate the peer relationship between DeviceA and DeviceC. This is because DeviceA can find the default route in the IP routing table based on the peer's IP address.
- If DeviceA and DeviceC establish an IBGP peer relationship, you are advised to enable BGP peer tracking on both devices to ensure that the peer relationship can be terminated soon after a fault occurs.
- If establishing a BGP peer relationship depends on IGP routes, you need to configure how long BGP peer tracking waits after detecting peer unreachability before it terminates the BGP connection. The configured length of time should be longer than the IGP route convergence time. Otherwise, before IGP route flapping caused by intermittent disconnection is rectified, the BGP peer relationship may have been terminated, resulting in unnecessary BGP convergence.

BGP 6PE

Background

With the wide application of IPv6 technologies, more and more separate IPv6 networks emerge. IPv6 provider edge (6PE), a technology designed to provide IPv6 services over IPv4 networks, allows service providers to provide IPv6 services

without constructing new IPv6 backbone networks. The 6PE solution connects separate IPv6 networks using MPLS tunnels on IPv4 networks. The 6PE solution implements IPv4/IPv6 dual stack on the PEs of Internet service providers (ISPs) and uses MP-BGP to assign labels to IPv6 routes. In this manner, the 6PE solution connects separate IPv6 networks over IPv4 tunnels between PEs.

Related Concepts

In real-world situations, different metro networks of a carrier or backbone networks of collaborative carriers often span different ASs. 6PE is classified as either intra-AS 6PE or inter-AS 6PE, depending on whether separate IPv6 networks connect to the same AS. If separate IPv6 networks are connected to different ASs, inter-AS 6PE can be implemented through inter-AS 6PE Option B (with ASBRs as PEs), inter-AS 6PE Option C, or inter-AS 6PE Option B mode.

- Intra-AS 6PE: Separate IPv6 networks are connected to the same AS. PEs in the AS exchange IPv6 routes through MP-IBGP peer relationships.
- Inter-AS 6PE Option B (with ASBRs as PEs): ASBRs in different ASs exchange IPv6 routes through MP-EBGP peer relationships.
- Inter-AS 6PE Option B: ASBRs in different ASs exchange IPv6 labeled routes through MP-EBGP peer relationships.
- Inter-AS 6PE Option C: PEs in different ASs exchange IPv6 labeled routes through multi-hop MP-EBGP peer relationships.

Intra-AS 6PE

Figure 1-342 shows intra-AS 6PE networking. 6PE runs on the edge of an ISP network. PEs that connect to IPv6 networks use the IPv4/IPv6 dual stack. PEs and CEs exchange IPv6 routes using the IPv6 EBGP or an IGP. PEs exchange routes with each other and with Ps using an IPv4 routing protocol. PEs need to establish tunnels to transparently transmit IPv6 packets. The tunnels mainly include MPLS label switched paths (LSPs) and MPLS Local Ifnet tunnels. By default, LSPs are preferentially selected. If no LSPs are available, MPLS Local Ifnet tunnels are used.

Figure 1-342 Intra-AS 6PE networking diagram

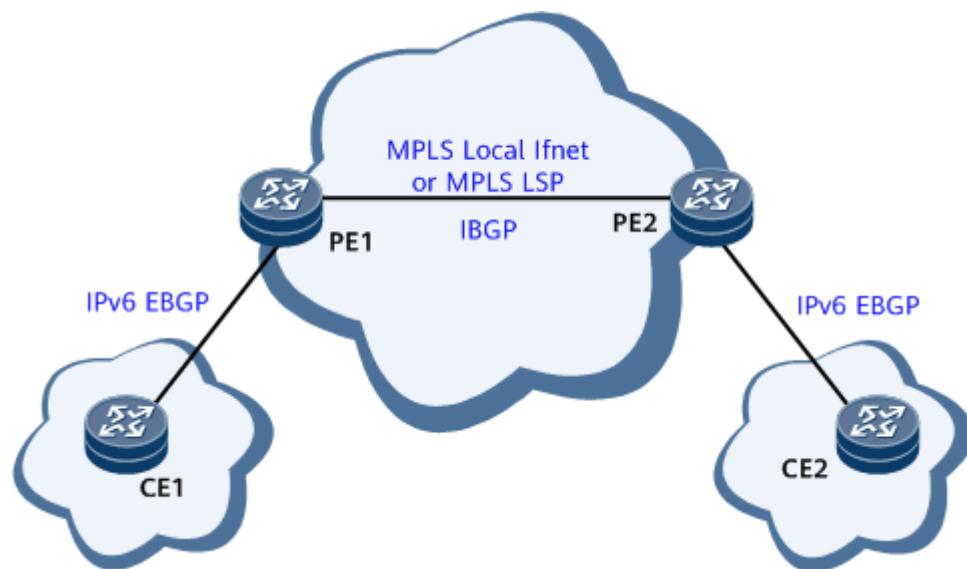


Figure 1-343 shows an intra-AS 6PE scenario where CE2 sends routes to CE1 and CE1 sends a packet to CE2. I-L indicates the inner label, whereas O-L indicates the outer tunnel label. The outer tunnel label is allocated by MPLS and is used to divert packets to the BGP next hop. The inner label indicates the outbound interface of the packets or the CE to which the packets belong.

The route transmission process in an intra-AS 6PE scenario is as follows:

1. CE2 sends an intra-AS IPv6 route to PE2, its EBGP peer.
2. Upon receipt of the IPv6 route from CE2, PE2 changes the next hop of the route to itself, assigns an inner label to the route, and sends the IPv6 labeled route to PE1, its IBGP peer.
3. Upon receipt of the IPv6 labeled route from PE2, PE1 recurses the route to a tunnel and delivers information about the route to the forwarding table. Then, PE1 changes the next hop of the route to itself, removes the label from the route, and sends the ordinary IPv6 route to CE1, its EBGP peer.

In this manner, the IPv6 route is transmitted from CE2 to CE1.

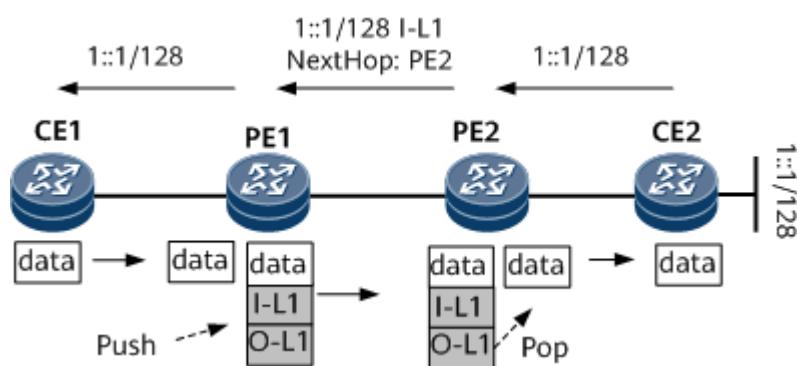
The packet transmission process in an intra-AS 6PE scenario is as follows:

1. CE1 sends an ordinary IPv6 packet to PE1 over a public network IPv6 link.
2. Upon receipt of the IPv6 packet from CE1, PE1 searches its forwarding table based on the destination address of the packet, encapsulates the packet with the inner label and outer tunnel label, and sends the IPv6 packet with two labels to PE2 over a public network tunnel.
3. Upon receipt of the IPv6 packet with two labels, PE2 removes the two labels and forwards the packet to CE2 over an IPv6 link based on the destination address of the packet.

In this way, the IPv6 packet is transmitted from CE1 to CE2.

The route and packet transmission processes show that CEs are unaware of whether the public network is an IPv4 or IPv6 network.

Figure 1-343 Route and packet transmission in an intra-AS 6PE scenario



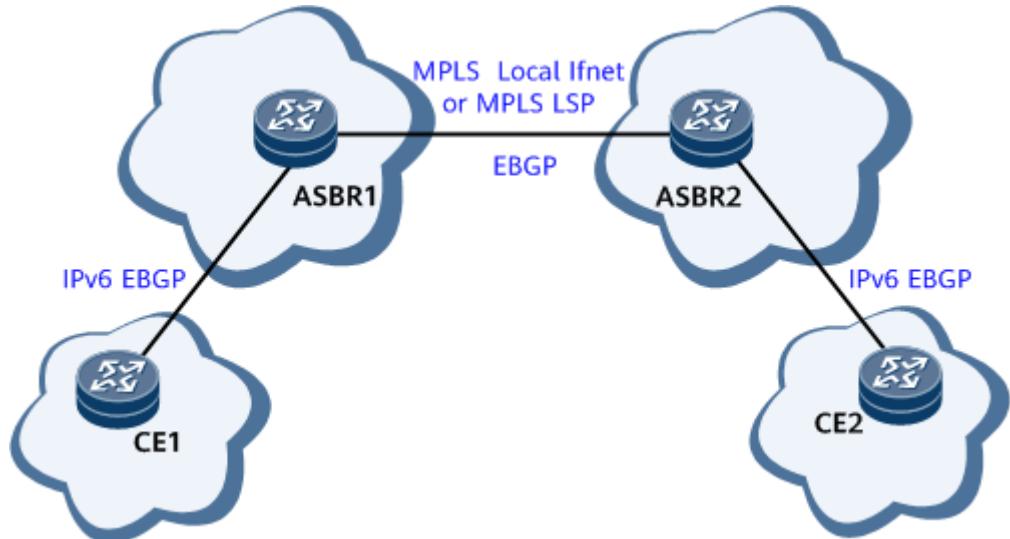
Inter-AS 6PE

- **Inter-AS 6PE Option B (with ASBRs as PEs)**

Figure 1-344 shows the networking of inter-AS 6PE Option B (with ASBRs as PEs). Inter-AS 6PE Option B (with ASBRs as PEs) is similar to intra-AS 6PE. The only difference is that in the former scenario, ASBRs (shown in **Figure 1-344**)

establish an EBGP peer relationship. The route and packet transmission processes in an inter-AS 6PE Option B scenario (with ASBRs as PEs) are also similar to those in an intra-AS 6PE scenario. For details, see [Figure 1-343](#).

Figure 1-344 Networking diagram for inter-AS 6PE Option B (with ASBRs as PEs)



- **Inter-AS 6PE Option B**

[Figure 1-345](#) shows inter-AS 6PE Option B networking. ASBRs exchange labeled routes with each other and with PEs using an IPv4 routing protocol. Tunnels must be established between each PE and the ASBR in the same AS and between ASBRs to transparently transmit IPv6 packets. The tunnels between ASBRs mainly include MPLS LSPs, MPLS Local Ifnet tunnels, GRE tunnels, and MPLS TE tunnels. By default, LSPs are preferentially selected. If no LSPs are available, MPLS Local Ifnet tunnels are used. To use MPLS TE or GRE tunnels, configure a tunnel policy on the ASBRs.

Figure 1-345 Networking diagram for inter-AS 6PE Option B

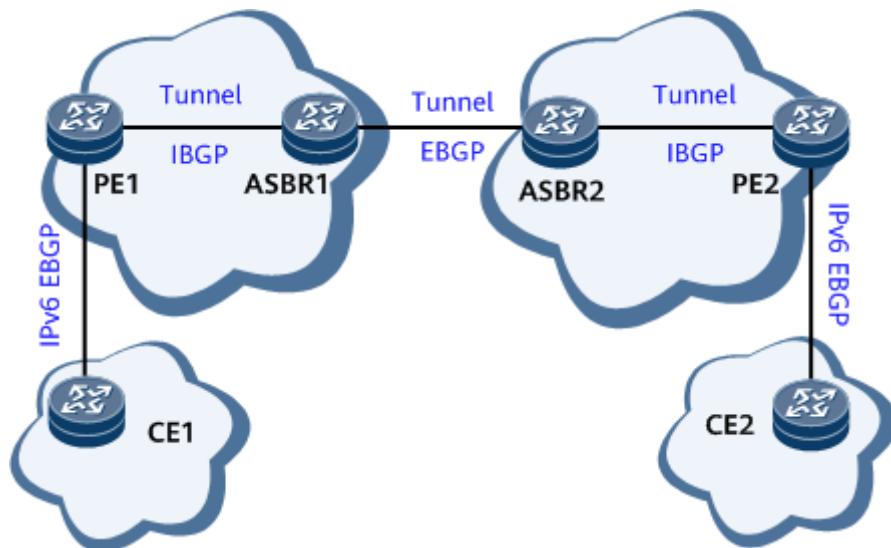
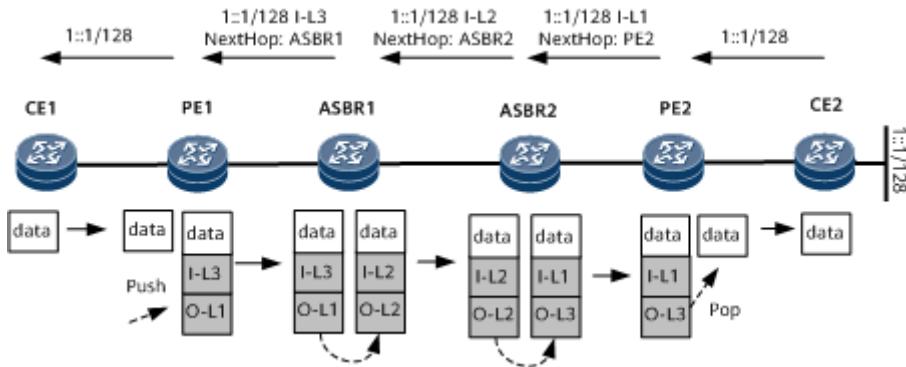


Figure 1-346 shows the inter-AS 6PE Option B scenario where CE2 sends routes to CE1 and CE1 sends packets to CE2. I-L indicates the inner label, whereas O-L indicates the outer tunnel label.

Figure 1-346 Route and packet transmission in an inter-AS 6PE Option B scenario



The route transmission process in an inter-AS 6PE Option B scenario is as follows:

- CE2 sends an intra-AS IPv6 route to PE2, its EBGP peer.
- Upon receipt of the IPv6 route from CE2, PE2 changes the next hop of the route to itself, assigns an inner label to the route, and sends the IPv6 labeled route to ASBR2, its IBGP peer.
- Upon receipt of the IPv6 labeled route from PE2, ASBR2 recurses the route to a tunnel and delivers the route to the forwarding table. Then, ASBR2 changes the next hop of the route to itself, allocates a new inner label to the route, and sends the route to ASBR1, its EBGP peer.
- Upon receipt of the IPv6 labeled route from ASBR2, ASBR1 recurses the route to a tunnel and delivers the route to the forwarding table. Then, ASBR1 changes the next hop of the route to itself, allocates a new inner label to the route, and sends the route to PE1, its IBGP peer.
- Upon receipt of the IPv6 labeled route from ASBR1, PE1 recurses the route to a tunnel and delivers the route to the forwarding table. Then, PE1 changes the next hop of the route to itself, removes the label from the route, and sends the ordinary IPv6 route to CE1, its EBGP peer.

In this manner, the IPv6 route is transmitted from CE2 to CE1.

The packet forwarding process in an inter-AS 6PE Option B scenario is as follows:

- CE1 sends an ordinary IPv6 packet to PE1 over a public network IPv6 link.
- Upon receipt of the IPv6 packet from CE1, PE1 searches its forwarding table based on the destination address of the packet, encapsulates the packet with the inner label and outer tunnel label, and sends the IPv6 packet with two labels to ASBR1 over an intra-AS public network LSP.
- Upon receipt of the packet from PE1, ASBR1 removes the two labels from the packet, searches its forwarding table based on the destination address of the packet, encapsulates the packet with a new inner label and outer tunnel label, and sends the IPv6 packet to ASBR2 over an inter-AS public network tunnel.

- d. Upon receipt of the packet from ASBR1, ASBR2 removes the two labels from the packet, searches its forwarding table based on the destination address of the packet, encapsulates the packet with a new inner label and outer tunnel label, and sends the IPv6 packet to PE2 over an intra-AS public network LSP.
- e. Upon receipt of the IPv6 packet with two labels, PE2 removes the two labels and forwards the packet to CE2 over an IPv6 link based on the destination address of the packet.

In this way, the IPv6 packet is transmitted from CE1 to CE2.

- **Inter-AS 6PE Option C**

Figure 1-347 shows inter-AS 6PE Option C networking. The difference between inter-AS 6PE Option B and inter-AS 6PE Option C is as follows: in an inter-AS 6PE Option C scenario, PEs establish a multi-hop MP-EBGP peer relationship, exchange labeled routes using an IPv4 routing protocol, and transparently transmit IPv6 packets over an end-to-end BGP LSP between the PEs.

 **NOTE**

Two inter-AS 6PE Option C solutions are available, depending on the establishment methods of end-to-end LSPs. In an inter-AS 6PE Option C scenario, PEs establish a multi-hop MP-EBGP peer relationship to exchange IPv6 labeled routes and establish an end-to-end BGP LSP to transmit IPv6 packets. The way in which the end-to-end BGP LSP is established does not matter much to inter-AS 6PE Option C and therefore is not described here.

Figure 1-347 Networking diagram for inter-AS 6PE Option C

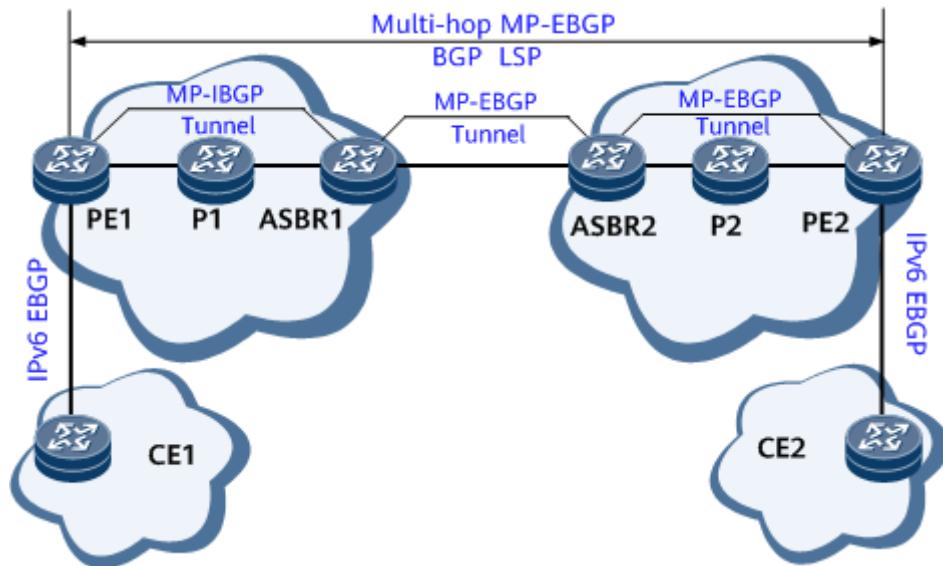


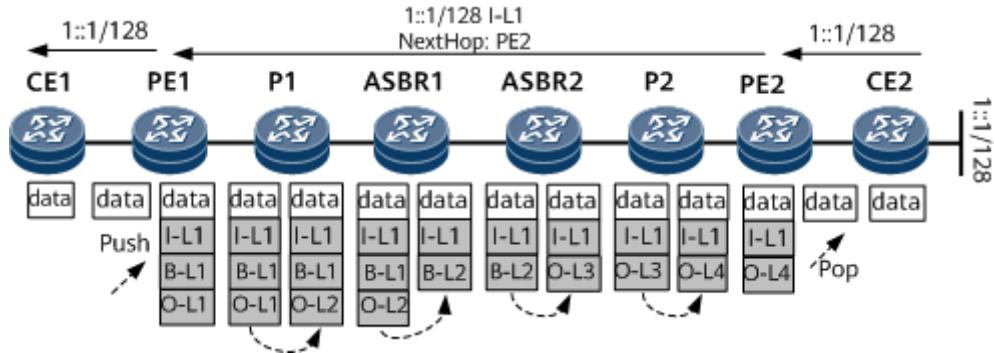
Figure 1-348 shows the inter-AS 6PE Option C scenario where CE2 sends routes to CE1 and CE1 sends packets to CE2. I-L indicates an inner label, B-L indicates a BGP LSP label, and O-L indicates an outer tunnel label.

 NOTE

In [Figure 1-348](#), the following two assumptions are made for clearer description:

- An MPLS local Ifnet tunnel is established between the two ASBRs.
- MPLS does not use the penultimate hop popping (PHP) function.

Figure 1-348 Route and packet transmission in an inter-AS 6PE Option C scenario



The route transmission process in an inter-AS 6PE Option C scenario is as follows:

- CE2 sends an intra-AS IPv6 route to PE2, its EBGP peer.
- Upon receipt of the IPv6 route from CE2, PE2 changes the next hop of the route to itself, assigns an inner label to the route, and sends the IPv6 labeled route to PE1, its MP-EBGP peer.
- Upon receipt of the IPv6 labeled route from PE2, PE1 recurses the route to a tunnel and delivers information about the route to the forwarding table. Then, PE1 changes the next hop of the route to itself, removes the label from the route, and sends the ordinary IPv6 route to CE1, its EBGP peer.

In this manner, the IPv6 route is transmitted from CE2 to CE1. During route transmission, ASBRs transparently transmit packets carrying IPv6 labeled routes without modifying the IPv6 labeled routes.

The packet forwarding process in an inter-AS 6PE Option C scenario is as follows:

- CE1 sends an ordinary IPv6 packet to PE1 over a public network IPv6 link.
- Upon receipt of the IPv6 packet from CE1, PE1 searches its forwarding table based on the destination address of the packet, changes the next hop of the packet, encapsulates the packet with an inner label, a BGP LSP label, and an outer tunnel label, and sends the IPv6 packet to P1 over an intra-AS public network tunnel.
- Upon receipt of the IPv6 packet from PE1, P1 removes the outer label, adds a new outer label to the packet, and forwards the packet with three labels to ASBR1 over an intra-AS public network tunnel.
- Upon receipt of the IPv6 packet from P1, ASBR1 removes the outer label and BGP LSP label, encapsulates a new BGP LSP label into the IPv6 packet, and forwards the IPv6 packet with two labels to ASBR2 over the inter-AS public network tunnel.

- e. Upon receipt of the IPv6 packet from ASBR1, ASBR2 removes the BGP LSP label, encapsulates the packet with a new outer label, and forwards the IPv6 packet with two labels to P2 over an intra-AS public network tunnel.
- f. Upon receipt of the IPv6 packet from ASBR2, P2 removes the outer label from the packet, encapsulates the packet with a new outer label, and forwards the packet with two labels to PE2 over an intra-AS public network tunnel.
- g. Upon receipt of the IPv6 packet with two labels, PE2 removes the two labels and forwards the packet to CE2 over an IPv6 link based on the destination address of the packet.

In this way, the IPv6 packet is transmitted from CE1 to CE2.

Usage Scenario

Each 6PE mode has its advantages and usage scenarios. Intra-AS 6PE applies to scenarios where separate IPv6 networks connect to the same AS. Inter-AS 6PE applies to scenarios where separate IPv6 networks connect to different ASs. [Table 1-106](#) lists the usage scenarios of inter-AS 6PE.

Table 1-106 Usage scenarios of inter-AS 6PE

Mode	Characteristic	Usage Scenario
Inter-AS 6PE Option B (with ASBRs as PEs)	Advantage: The configuration is simple and similar to that for intra-AS 6PE, and additional inter-AS configuration is not required. Disadvantage: The scalability is poor. ASBRs must manage information about all IPv6 labeled routes, which increases the performance requirements on the ASBRs.	It applies to small networks where different IPv6 networks connect to ASBRs in different ASs. This mode is especially applicable to the scenario where a small number of ASs are spanned.

Mode	Characteristic	Usage Scenario
Inter-AS 6PE Option B	<p>Advantage: MPLS tunnels are established segment by segment, reducing management costs.</p> <p>Disadvantage: Information about IPv6 labeled routes is stored and advertised by ASBRs in different ASs. When a large number of IPv6 labeled routes exist, the ASBRs are overburdened and are likely to become fault points.</p>	It applies to an inter-AS Option B public network with multi-segment tunnels established between PEs in different ASs that are connected to separate IPv6 networks.
Inter-AS 6PE Option C	<p>Advantage: IPv6 labeled routes are directly exchanged between the ingress and egress PEs and do not need to be stored or forwarded by transit devices.</p> <p>Information about IPv6 labeled routes is managed by PEs only, and ASBRs are no longer route capacity bottlenecks.</p> <p>Disadvantage: Maintaining an end-to-end BGP LSP connection is costly.</p>	<p>It applies to an inter-AS Option C public network with E2E tunnels established between PEs in different ASs that are connected to separate IPv6 networks.</p> <p>This solution is recommended when multiple ASs are spanned.</p>

Benefits

6PE offers the following benefits:

- **Easy maintenance:** All configurations are performed on PEs, and IPv6 networks are unaware of the IPv4 network. The existing IPv4 network is used to carry IPv6 services, simplifying network maintenance.
- **Low network construction cost:** Carriers can make full use of the existing MPLS network resources and provide IPv6 services for users without upgrading the network. 6PE devices can also provide various types of services, such as IPv6 VPN and IPv4 VPN services.

BGP ORF

Outbound route filtering (ORF) enables a device to send local routing policies to a peer, which can then use the received policies to filter out unwanted routes before route advertisement.

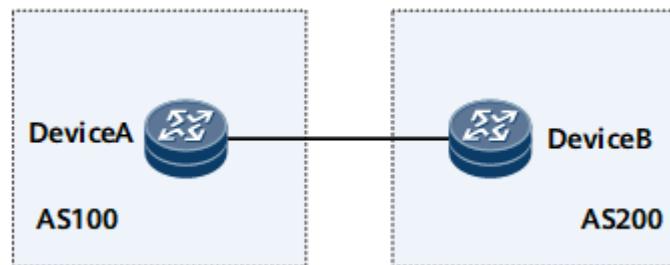
In most cases, users expect a carrier to send only the routes that they require, and the carrier does not want to maintain a separate export routing policy for each user. ORF meets the requirements of both users and the carrier. ORF supports on-demand route advertisement, which significantly reduces bandwidth consumption and the configuration workload.

Prefix-based ORF, defined in standard protocols, can be used to send prefix-based import policies configured by users to a carrier through Route-refresh messages. The carrier's device then filters out unwanted routes before route advertisement based on the received policies. This prevents the users from receiving a large number of unwanted routes and thus conserves resources.

Application

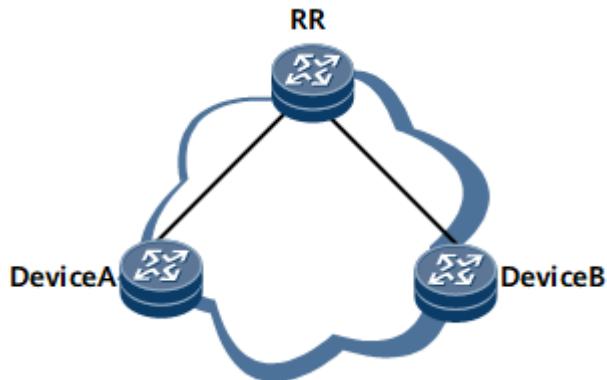
On the network shown in [Figure 1-349](#), after directly connected BGP peers DeviceA and DeviceB negotiate the prefix-based ORF capability, DeviceA adds the locally configured prefix-based import policy to a Route-Refresh message and sends the message to DeviceB. After receiving the Route-Refresh message, DeviceB constructs an export policy based on the message and sends routes to DeviceA accordingly.

Figure 1-349 Directly connected BGP peers



[Figure 1-350](#) shows an AS with an RR and BGP peers, and DeviceA and DeviceB are clients of the RR. DeviceA and DeviceB negotiate the prefix-based ORF capability with the RR, add their locally configured prefix-based import policies to a Route-Refresh message, and send the message to the RR. Based on the received prefix information, the RR constructs export policies, which are used to reflect routes to DeviceA and DeviceB.

Figure 1-350 AS with an RR



BGP Auto FRR

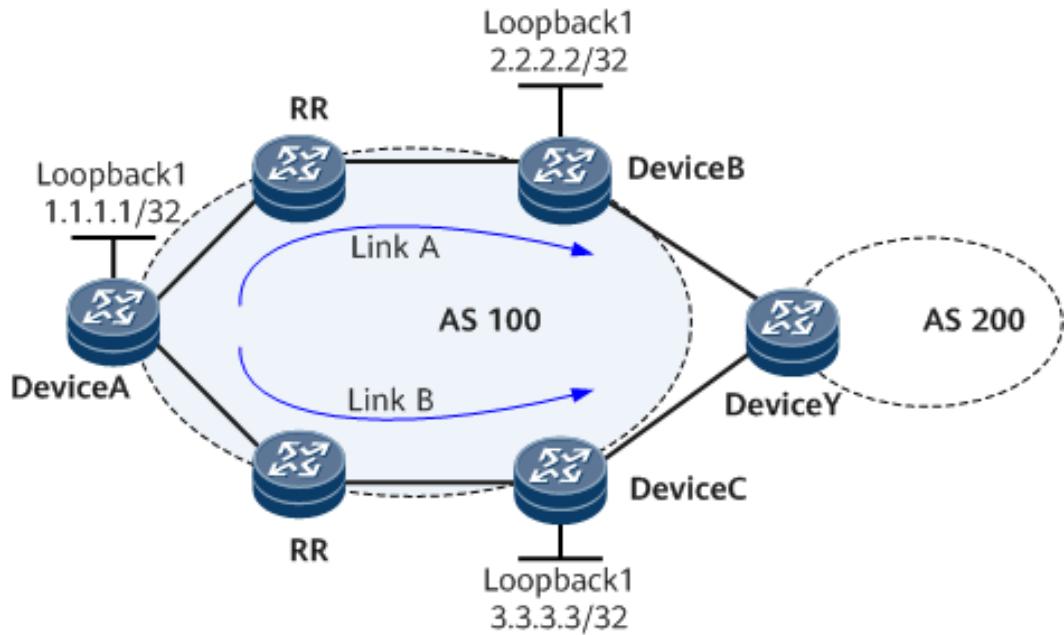
As a protection measure against link failures, BGP Auto fast reroute (FRR) is applicable to networks with primary and backup links. With BGP Auto FRR, traffic can be switched between two BGP peers or two next hops within sub-seconds.

With BGP Auto FRR, if a peer has multiple routes with the same prefix that are learned from different peers, it uses the optimal route as the primary link to forward packets and the sub-optimal route as a backup link. If the primary link fails, the peer rapidly notifies other peers that the BGP route has become unreachable and then switches traffic from the primary link to the backup link.

Application

On the network shown in [Figure 1-351](#), DeviceY advertises a learned BGP route to DeviceB and DeviceC in AS 100; DeviceB and DeviceC then advertise the route to the corresponding RR, which then reflects the route to DeviceA. In this case, DeviceA receives two routes whose next hops are DeviceB and DeviceC. Then, DeviceA selects a route based on a configured routing policy. Assume that the route sent by DeviceB is preferred. The route received through Link B functions as a backup link.

Figure 1-351 Network diagram of BGP Auto FRR



If a node along Link A fails or a fault occurs on Link A, the next hop of the route from DeviceA to DeviceB becomes unavailable. If Auto FRR is enabled on DeviceA, the forwarding plane then quickly switches DeviceA-to-DeviceY traffic to Link B, which ensures uninterrupted traffic transmission. In addition, DeviceA performs route reselection based on prefixes. Consequently, it selects the route sent by DeviceC and then updates its FIB.

BGP Dynamic Update Peer-Group

As the routing table increases in size and the network topology increases in complexity, BGP needs to be able to support more peers. When the router needs to send a large number of routes to many BGP peers and most of the peers share the same configuration, if the router groups each route and then send the route to each peer, the efficiency is low.

To improve the efficiency of route advertisement, BGP uses the dynamic update peer-group mechanism. The BGP peers with the same configurations are placed in an update peer-group. These routes are grouped once and then sent to all peers in the update peer-group, improving the grouping efficiency exponentially.

Without the update peer-group feature, a route to be sent has to be grouped separately for each peer. With the update peer-group feature, routes are grouped uniformly and then sent separately. Each route to be sent is grouped only once and then sent to all peers in the update peer-group, which improves the grouping efficiency and forwarding performance exponentially. When a large number of peers and routes exist, BGP dynamic update peer-groups greatly improve the BGP route grouping and forwarding performance.

Usage Scenario

The BGP dynamic update peer-groups feature is applicable to the following scenarios:

- Scenario with an international gateway
- Scenario with an RR
- Scenario where routes received from EBGP peers need to be sent to all IBGP peers

Figure 1-352 Networking for the international gateway

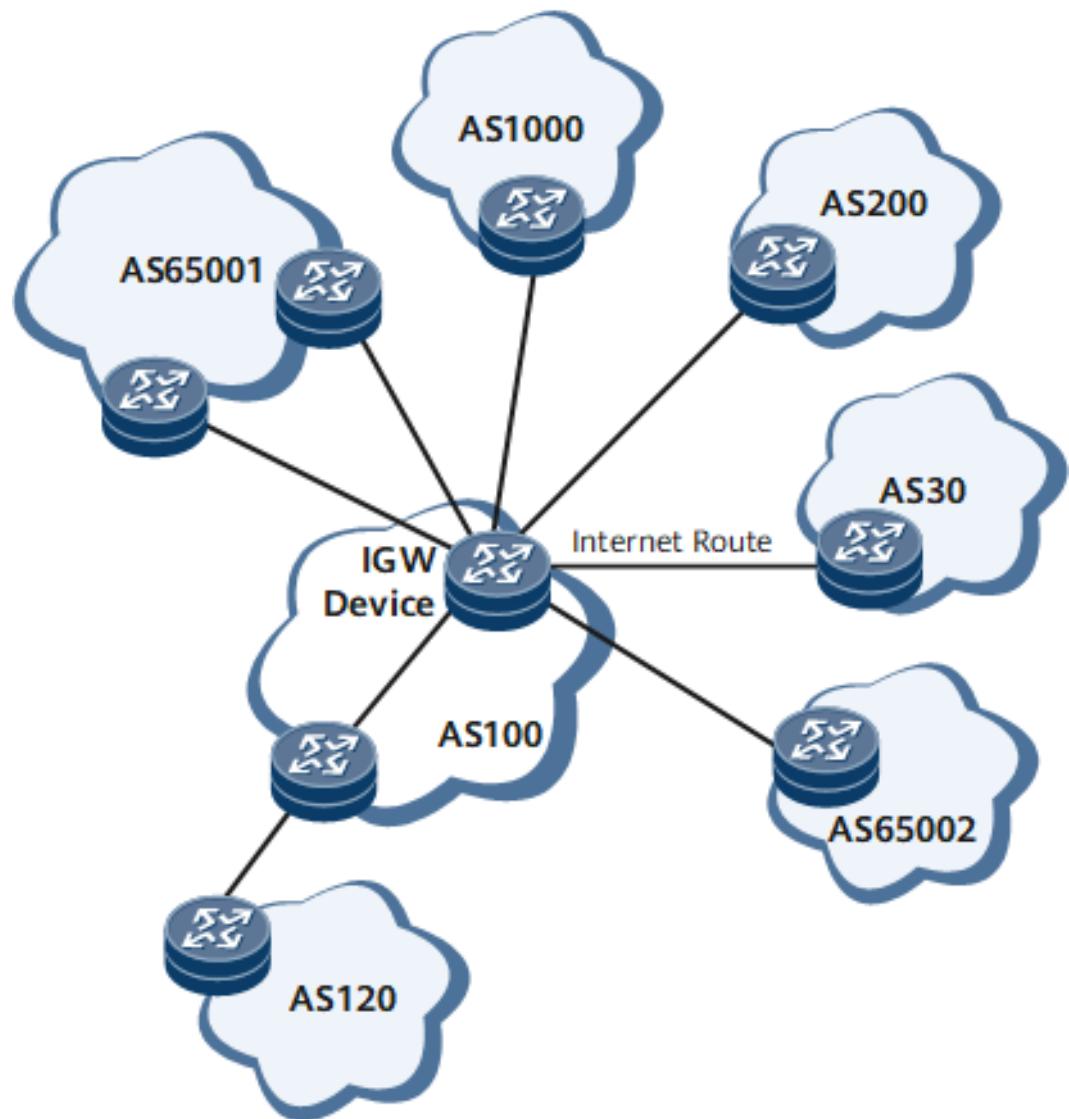


Figure 1-353 Networking for RRs with many clients

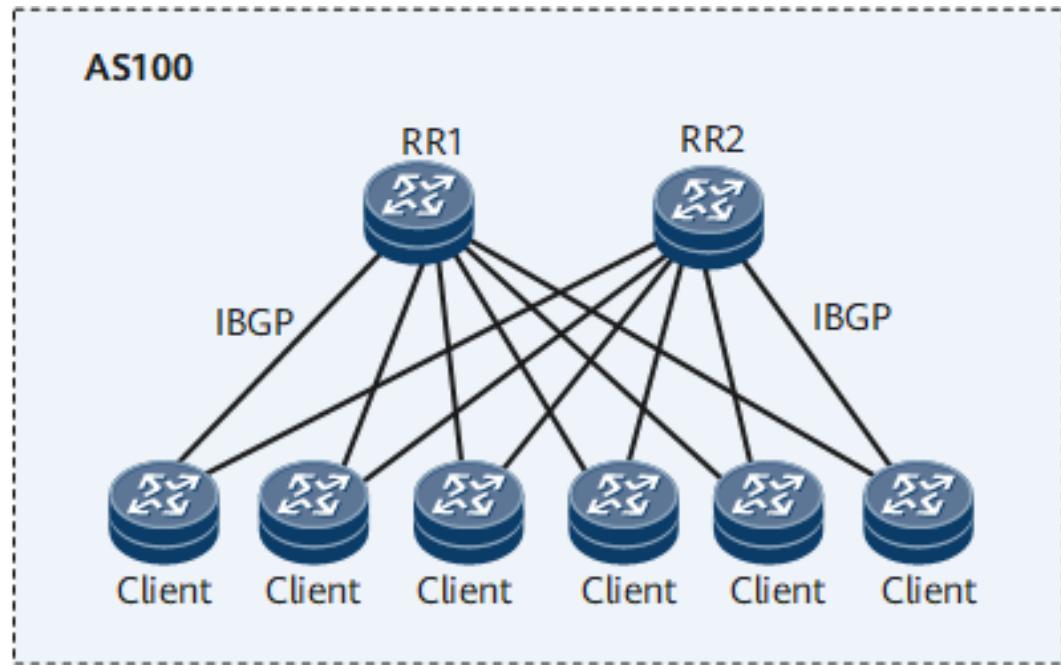
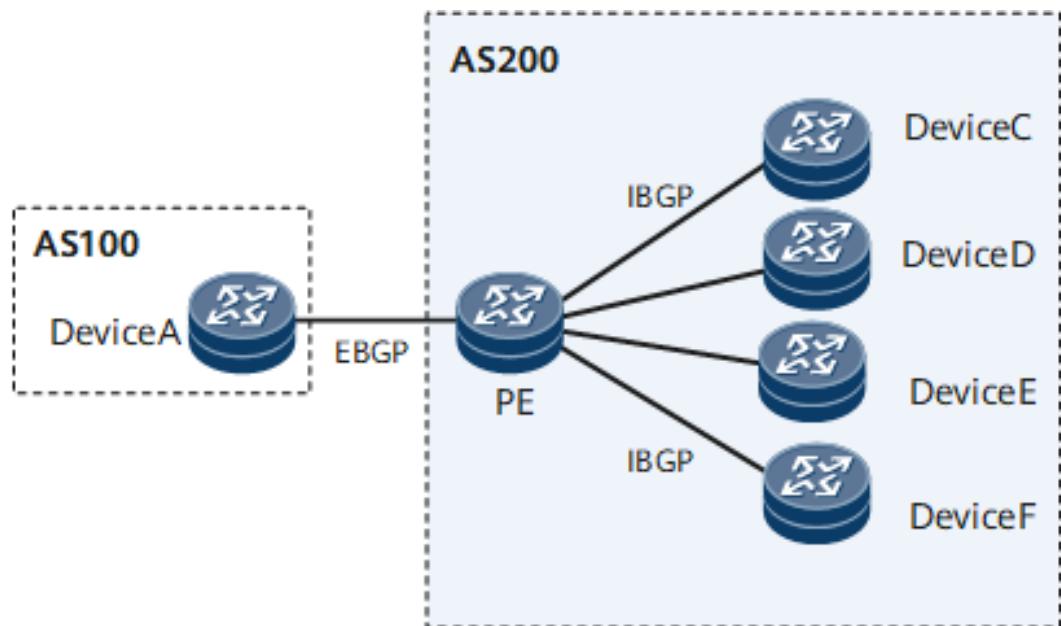


Figure 1-354 Networking for a PE connected to multiple IBGP peers



The preceding scenarios have in common that the router needs to send routes to a large number of BGP peers, most of which share the same configuration. This situation is most evident in the networking shown in [Figure 1-353](#). In addition, when there are a large number of peers and routes, the packet sending efficiency is a performance bottleneck.

The update peer-group feature can overcome this bottleneck. After the feature is applied, each routing update is grouped only once, and the generated Update

message is sent to all peers in the group. For example, an RR has 100 clients and needs to reflect 100,000 routes to them. If the RR groups routing updates per peer, it needs to group 100,000 routing updates 100 times (a total of 10 million) before sending Update messages to the 100 clients. The update peer-group feature improves performance 100-fold, because the 100,000 routing updates need to be grouped only once.

Distributed BGP

Distributed BGP allows BGP peers to be deployed in different BGP processes, improving both the BGP peer reliability and the route sending and receiving performance.

Usage Scenario

In traditional BGP, all BGP peers are deployed in one BGP process, which runs on the main control board. This can be referred to as centralized BGP. With the rapid development of networks, centralized BGP has the following characteristics:

- There are a large number of peers.
- Route flapping occurs frequently.
- BGP import or export policies are frequently modified, causing routes to be re-learned or re-advertised.

In distributed BGP, BGP peers are deployed in different BGP processes. Deploying distributed BGP offers the following benefits:

- Minimized impact on the entire BGP services when a single BGP process fails.
- Improved BGP peer stability: The processing capability of a single process is limited. Because BGP peers need to frequently advertise Keepalive and Update messages, the processing pressure of the BGP process increases and peer flapping occurs (due to delays in sending Keepalive messages) as the number of BGP peers increases. Distributed BGP uses multiple processes to balance BGP services that would otherwise run on a single process. Therefore, distributed BGP effectively reduces the message processing pressure on a single process, and deploying peers in different BGP processes improves BGP peer stability.
- Improved route sending and receiving efficiency: Distributed BGP uses multiple processes to parse messages during route receiving and to package routes by group during route sending. In addition, because import and export policies are based on peers, distributed BGP allows routes to be filtered against the policies in independent processes. This speeds up route sending and receiving.

Distributed BGP improves multi-peer stability and speeds up route learning when a large number of routes with the same prefix exist. However, after peers are deployed in distributed instances, additional system memory is consumed (about 5% to 15% of the memory occupied by BGP) to synchronize the BGP routing tables between distributed and base instances. Here are some suggestions on the deployment of distributed instances:

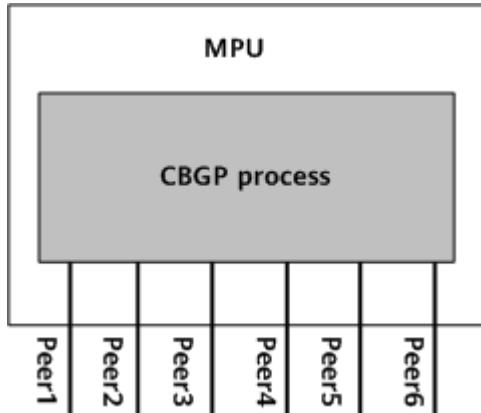
- If the system memory usage exceeds 70%, you are advised not to add distributed BGP instances.

- You are advised to deploy distributed BGP instances only when a large number of peers exist because distributed BGP is advantageous only when there are many peers.

Implementation

The architecture of traditional centralized BGP is shown in [Figure 1-355](#). In this architecture, all BGP services run in the same process, and all BGP peers directly establish connections with central BGP (C-BGP).

Figure 1-355 Architecture of centralized BGP

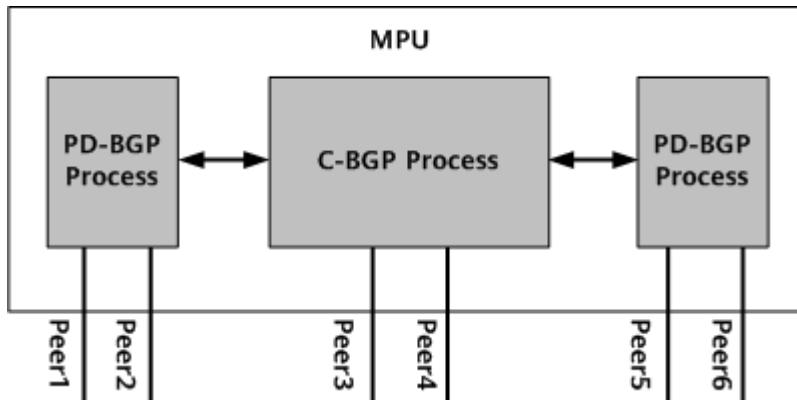


[Figure 1-356](#) shows the architecture of distributed BGP, which consists of C-BGP and peer-based distributed BGP (PD-BGP) processes. In this architecture, the C-BGP process transfers some BGP peers to PD-BGP processes. The number of PD-BGP processes can be planned based on the number of BGP peers.

NOTE

Peers of distributed BGP are unaware of whether distributed BGP or centralized BGP is used.

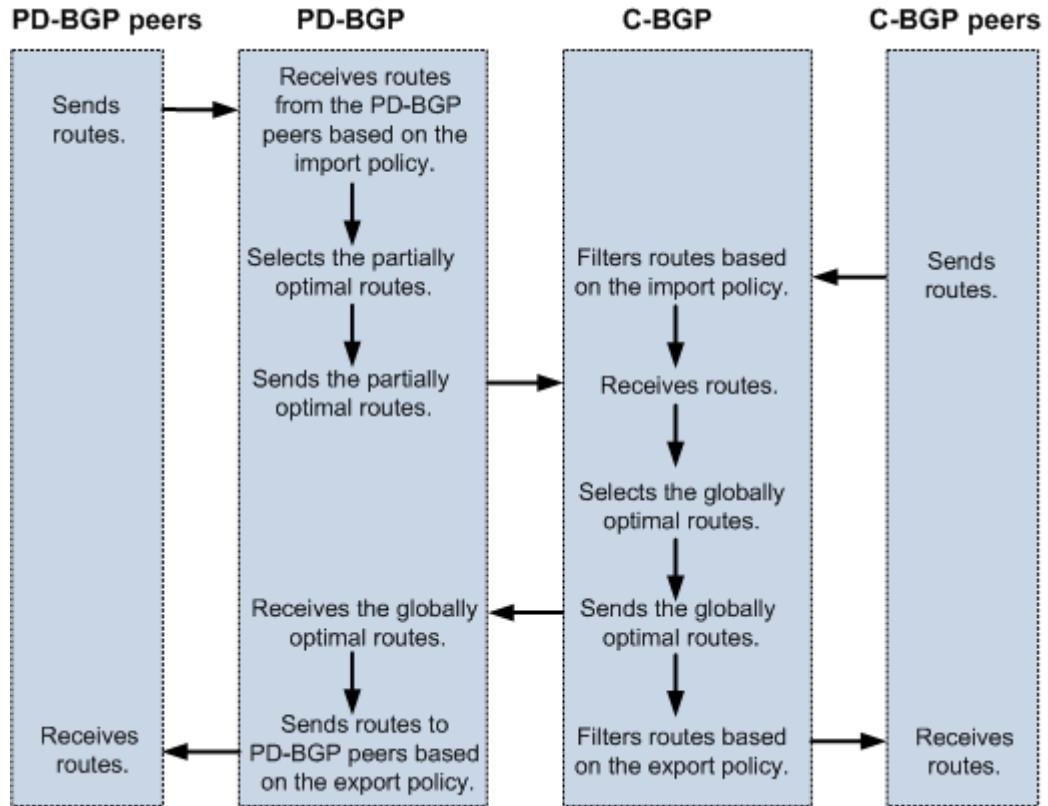
Figure 1-356 Architecture of distributed BGP



[Figure 1-357](#) shows how PD-BGP and C-BGP collaborate. After distributed BGP is enabled, C-BGP peer relationships can still be established. In addition, routes can be exchanged between C-BGP peers and between PD-BGP peers through import

and export policies, but not between C-BGP and PD-BGP peers through import and export policies.

Figure 1-357 Fundamentals of distributed BGP



The following conclusions can be reached according to [Figure 1-357](#):

- If a large number of routes with the same prefix are received from different peers, PD-BGP selects and sends the partially optimal routes to C-BGP. C-BGP then selects the globally optimal routes.
- PD-BGP and C-BGP send routes to and receive routes from PD-BGP peers and C-BGP peers, respectively, based on routing policies.

4-Byte AS Number

Purpose

2-byte autonomous system (AS) numbers used on networks range from 1 to 65535, and the available AS numbers are close to exhaustion as networks expand. Therefore, the AS number range needs to be extended. 4-byte AS numbers ranging from 1 to 4294967295 can address this problem. New speakers that support 4-byte AS numbers can co-exist with old speakers that support only 2-byte AS numbers.

Definition

4-byte AS numbers are extended from 2-byte AS numbers. Border Gateway Protocol (BGP) peers use a new capability code and optional transitive attributes

to negotiate the 4-byte AS number capability and transmit 4-byte AS numbers. This mechanism enables communication between new speakers and between old speakers and new speakers.

To support 4-byte AS numbers, an open capability code 0x41 is defined in a standard protocol for BGP connection negotiation. 0x41 indicates that the BGP speaker supports 4-byte AS numbers.

The following new optional transitive attributes are defined by standard protocols and used to transmit 4-byte AS numbers in old sessions:

- **AS4_Path** coded 0x11
- **AS4_Aggregator** coded 0x12

If a new speaker with an AS number greater than 65535 communicates with an old speaker, the old speaker needs to set the peer AS number to **AS_TRANS**. AS_TRANS is a reserved AS number with the value being 23456.

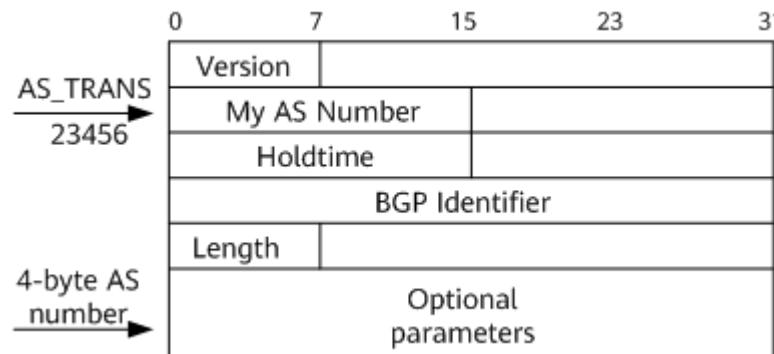
Related Concepts

- New speaker: a BGP peer that supports 4-byte AS numbers
- Old speaker: a BGP peer that does not support 4-byte AS numbers
- New session: a BGP connection established between new speakers
- Old session: a BGP connection established between a new speaker and an old speaker, or between old speakers

Fundamentals

BGP speakers negotiate capabilities by exchanging Open messages. [Figure 1-358](#) shows the format of Open messages exchanged between new speakers. The header of a BGP Open message is fixed, in which My AS Number is supposed to be the local AS number. However, My AS Number carries only 2-byte AS numbers, and does not support 4-byte AS numbers. Therefore, a new speaker adds the AS_TRANS 23456 to My AS Number and its local AS number to Optional Parameters before it sends an Open message to a peer. After the peer receives the message, it can determine whether the new speaker supports 4-byte AS numbers by checking Optional Parameters in the message.

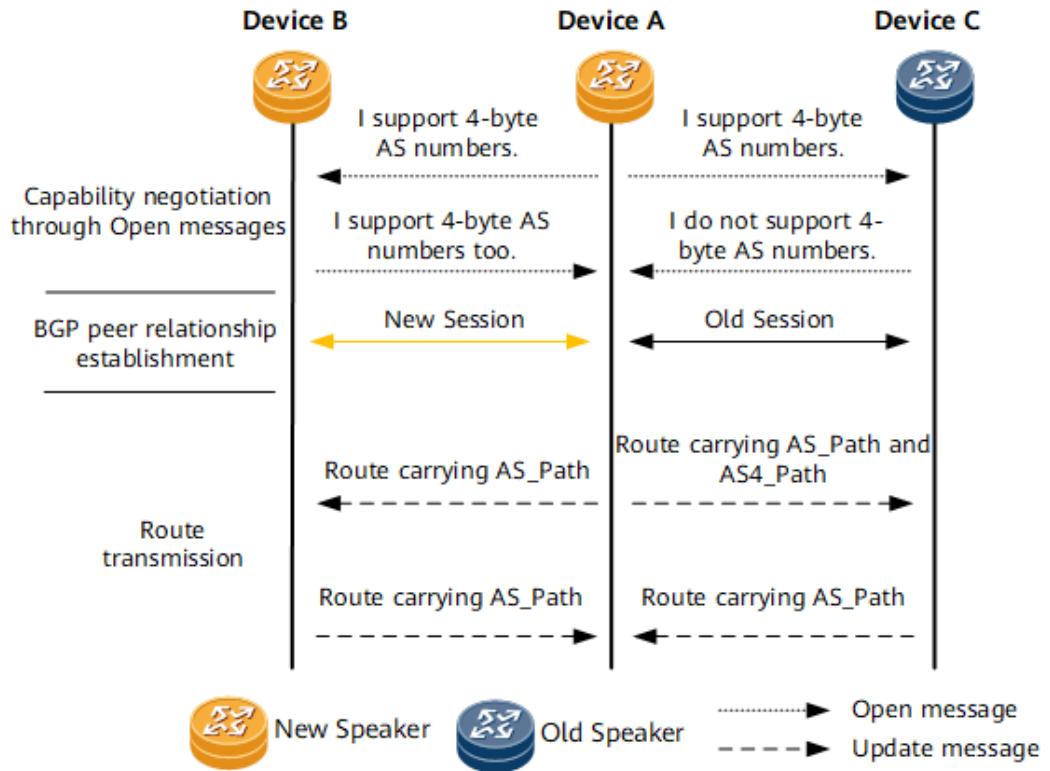
[Figure 1-358](#) Format of Open messages sent by new speakers



[Figure 1-359](#) shows how peer relationships are established between new speakers, and between an old speaker and a new speaker. BGP speakers notify

each other of whether they support 4-byte AS numbers by exchanging Open messages. After the capability negotiation, new sessions are established between new speakers, and old sessions are established between a new speaker and an old speaker.

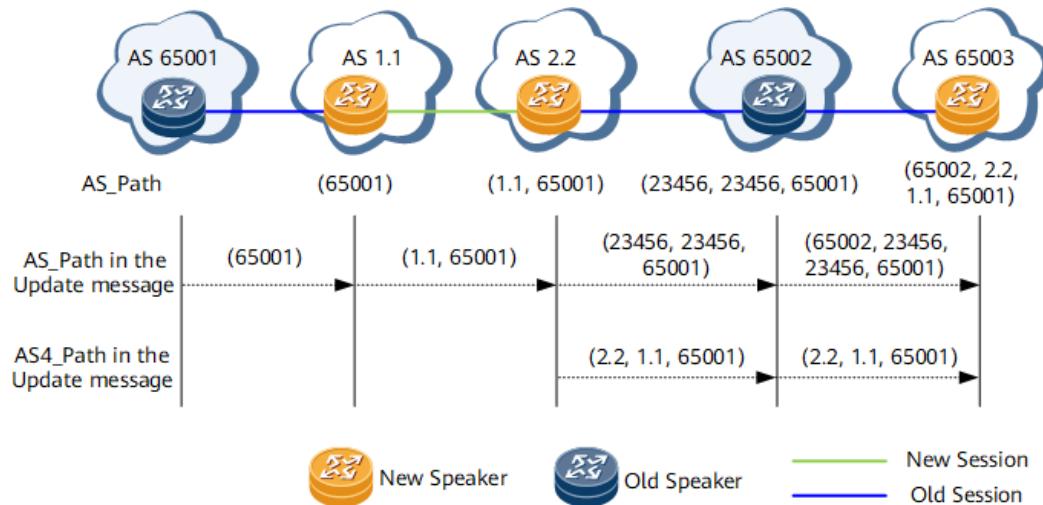
Figure 1-359 Process of establishing a BGP peer relationship



AS_Path and **Aggregator** in Update messages exchanged between new speakers carry 4-byte AS numbers, whereas **AS_Path** and **Aggregator** in Update messages sent by old speakers carry 2-byte AS numbers.

- When a new speaker sends an Update message carrying an AS number greater than 65535 to an old speaker, the new speaker uses **AS4_Path** and **AS4_Aggregator** to assist **AS_Path** and **AS_Aggregator** in transferring 4-byte AS numbers. **AS4_Path** and **AS4_Aggregator** are transparent to the old speaker. In the networking shown in [Figure 1-360](#), before the new speaker in AS 2.2 sends an Update message to the old speaker in AS 65002, the new speaker replaces each 4-byte AS number (2.2, 1.1, 65001) with 23456 in **AS_Path**. Therefore, the **AS_Path** carried in the Update message is (23456, 23456, 65001), and the carried **AS4_Path** is (2.2, 1.1, 65001). After the old speaker in AS 65002 receives the Update message, it transparently transmits the message to other ASs.
- When the new speaker receives an Update message carrying **AS_Path**, **AS4_Path**, **AS_Aggregator**, and **AS4_Aggregator** from the old speaker, the new speaker uses the reconstruction algorithm to reconstruct the actual **AS_Path** and **AS_Aggregator**. On the network shown in [Figure 1-360](#), after the new speaker in AS 65003 receives an Update message carrying **AS_Path** (65002, 23456, 23456, 65001) and **AS4_Path** (2.2, 1.1, 65001) from the old speaker in AS 65002, the new speaker reconstructs the actual **AS_Path** (65002, 2.2, 1.1, 65001).

Figure 1-360 Process of transmitting a BGP Update message



Format of 4-byte AS numbers

A 4-byte AS number can be an integer or in dotted notation. The system stores 4-byte AS numbers as unsigned integers, regardless of their formats. 4-byte AS numbers in dotted notation are in the format of X.Y. The formula of the conversion between 4-byte AS numbers for the two formats is as follows: Integer 4-byte AS number = X x 65536 + Y. For example, the 4-byte AS number in dotted notation **2.3** can be converted to the integer 4-byte AS number 131075 (2 x 65536 + 3).

The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X supports 4-byte AS numbers of both formats. The 4-byte AS numbers displayed in the configuration files are in the format configured by users.

By default, the 4-byte AS numbers displayed in the **display** and **debugging** command outputs are in dotted notation, regardless of the configured format. If the default display format of 4-byte AS numbers is changed from dotted notation to integer using a command, the 4-byte AS numbers will be displayed as integers automatically.

NOTICE

If you adjust the display format of 4-byte AS numbers, the matching results of AS_Path regular expressions and extended community filters are affected. Specifically, if the display format of 4-byte AS numbers is changed when an AS_Path regular expression or extended community filter is used as an export or import policy, the AS_Path regular expression or extended community filter needs to be reconfigured. If reconfiguration is not performed, routes cannot match the export or import policy, and a network fault occurs.

Benefits

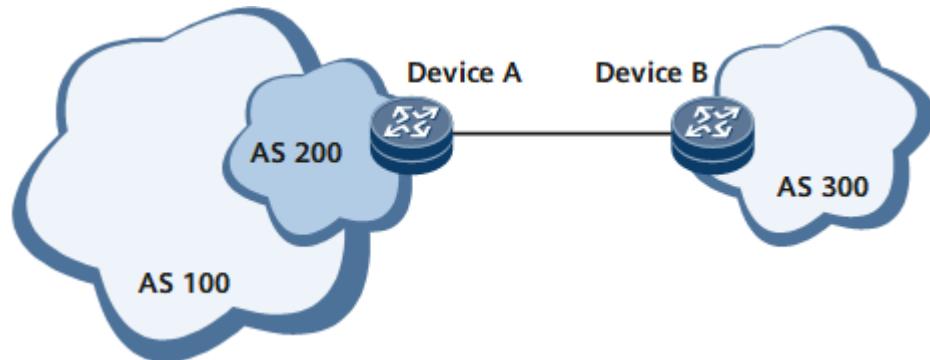
4-byte AS numbers alleviate AS number exhaustion and therefore are beneficial to carriers who need to expand the network scale.

Fake AS Number

In the acquisition and merger scenarios between carriers, if the acquiree and the acquirer are located in different ASs, the BGP peers of the acquiree need to be migrated to the AS of the acquirer. However, during network migration, the customer of the acquiree may not want to have local BGP configurations modified right away or at all. As a result, the BGP peer relationships may be interrupted for a long time.

In **Figure 1-361**, the AS number of carrier A is 100, whereas the AS number of carrier B is 200. Device A belongs to carrier B. Then carrier A acquires carrier B. In this case, the AS number of device A needs to be changed from 200 to 100. Because device A already has a BGP peer relationship established with device B in AS 300 using AS 200, device A's AS number used to establish the BGP peer relationship needs to be changed to 100. The carrier of AS 100 and the carrier of AS 300 then need to communicate about the change. In addition, the AS number configured on device A and peer AS number configured on device B may not be changed at the same time, which will lead to a lengthy interruption of the BGP peer relationship between the two devices. To ensure a smooth merger, you can run the **peer fake-as** command on device A to set AS 200 of carrier B as a fake AS number so that device A's AS number used to establish the BGP peer relationship between devices A and B does not need to be changed.

Figure 1-361 Network 1 with a fake AS number

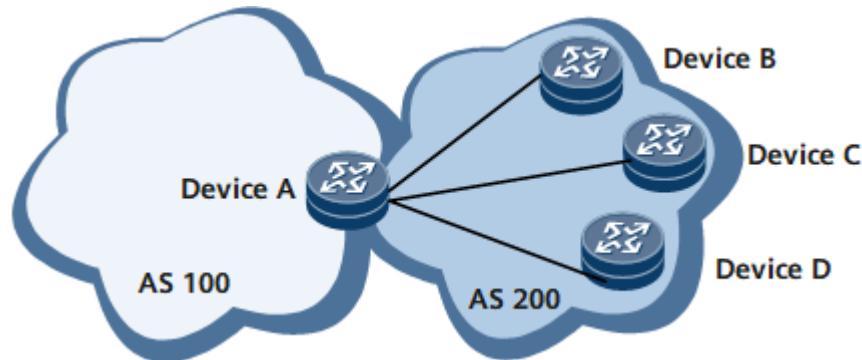


In addition, the AS number of the original BGP speakers of carrier B may be changed to the actual AS number at any time when BGP peer relationships are established with devices of carrier A after the merger. If carrier B has a large number of BGP speakers and some of the speakers use the actual AS number whereas other speakers use the fake AS number during BGP peer relationship establishment with devices of carrier A, the local configuration on BGP speakers of carrier B needs to be changed based on the configuration of the peer AS number, which increases the workload of maintenance. To address this problem, you can run the **peer fake-as** command with **dual-as** specified to allow the local end to use the actual or fake AS number to establish a BGP peer relationship with the specified peer.

In **Figure 1-362**, the AS number of carrier A is 100, whereas the AS number of carrier B is 200; devices A, B, C, and D belong to carrier B, and device A establishes an IBGP peer relationship with device B, device C, and device D each. Then carrier A acquires carrier B. In this case, the AS number of device A needs to be changed from 200 to 100. Because the AS number used by device A to establish the IBGP peer relationship with devices B, C, and D is 200, the AS number needs to be

changed to 100. In this case, carrier A and carrier B need to communicate about the change. In addition, the AS number configured on device A and peer AS number configured on devices B, C, and D may not be changed at the same time, which will lead to a lengthy interruption of the IBGP peer relationships. To ensure a smooth merger, you can run the **peer fake-as** command on device A to set AS 200 of carrier B as a fake AS number so that device A's AS number used to establish the IBGP peer relationships with devices B, C, and D does not need to be changed.

Figure 1-362 Network 2 with a fake AS number



BMP

Background

The BGP Monitoring Protocol (BMP) can monitor the BGP running status and trace data of BGP routes on network devices in real time. The BGP running status includes the establishment and termination of peer relationships and update of routing information. The trace data of BGP routes indicates how BGP routes on a device are processed; for example, processing of the routes that match an import or export route-policy.

Before BMP is implemented, only manual query can be used to obtain the BGP running status of devices, resulting in low monitoring efficiency.

After BMP is implemented, a device can be connected to a BMP server and configured to report its BGP running status to the server, significantly improving monitoring efficiency.

BMP Message Types

BMP sessions include Initiation, PU, RM, PD, SR, Termination, and ROFT messages, which are sent in packets. The information reported in these messages mainly includes BGP routing information, BGP peer information, and device vendor and version information.

- Initiation message: sent by a monitored device to the monitoring server to report information such as the device vendor and software version.
- Peer Up Notification (PU) message: sent by a monitored device to notify the monitoring server that a BGP peer relationship has been established.

- Route Monitoring (RM) message: used to provide the monitoring server with a collection of all routes received from a BGP peer and notify the server of route addition or withdrawal in real time.
- Peer Down Notification (PD) message: sent to notify the monitoring server that a BGP peer relationship has been disconnected.
- Stats Reports (SR) message: sent to report statistics about the device running status to the monitoring server.
- Termination message: sent to report the cause for closing a BMP session to the monitoring server.
- Route Policy and Attribute Trace (ROFT) message: used to report the trace data of routes to the monitoring server in real time.

 NOTE

BMP sessions are unidirectional. Devices send messages to the monitoring server but ignore messages sent by the server.

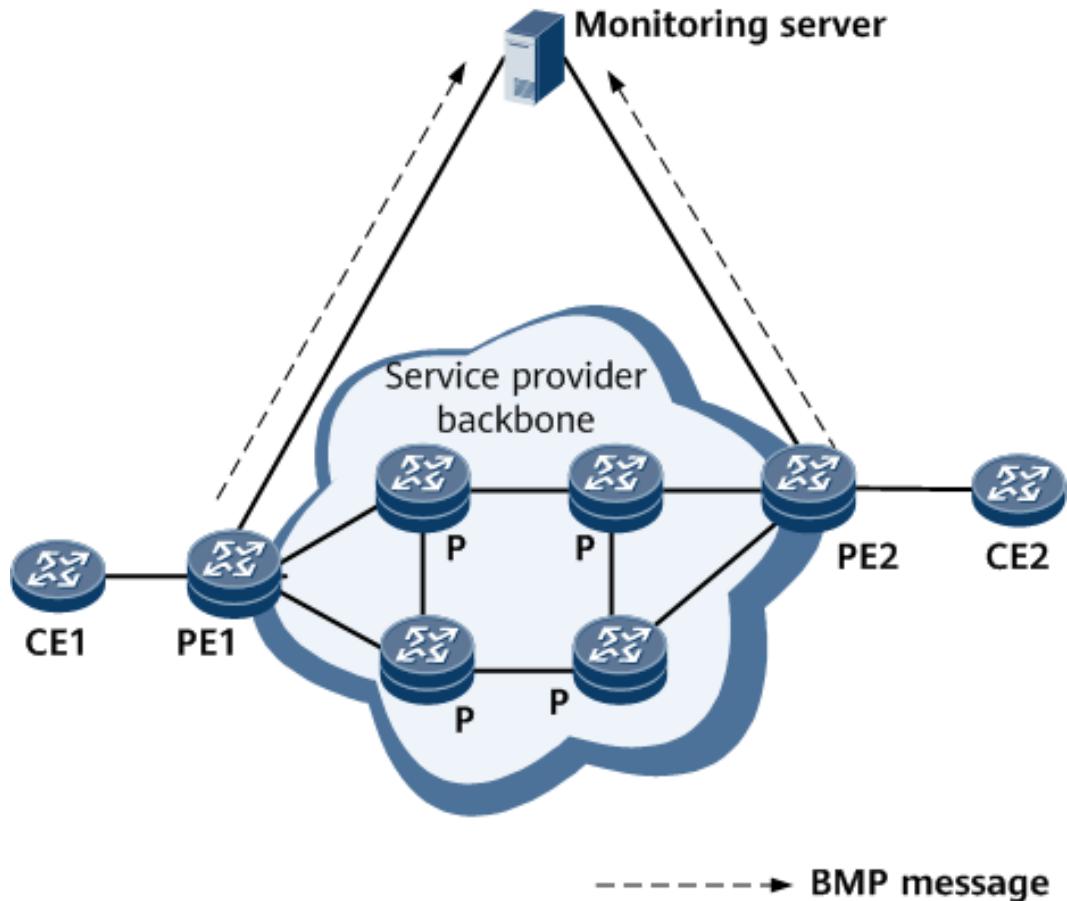
Implementation

On the network shown in [Figure 1-363](#), TCP connections are established between the monitoring server and monitored devices (PE1 and PE2 shown in the figure). The monitored devices send unsolicited BMP messages to the monitoring server to report information about the BGP running status. Upon receiving these BMP messages, the monitoring server parses them and displays the BGP running status in the monitoring view. By analyzing the headers in the received BMP messages, the monitoring server can determine which BGP peers have advertised the routes carried in these messages.

When establishing a connection between a BGP device and monitoring server, note the following guidelines:

- BMP operates over TCP, and you can specify a port number for the TCP connection between the BGP device and monitoring server.
- One device can connect to multiple monitoring servers, and one monitoring server can also connect to multiple devices.
- Each BMP instance can connect to multiple monitoring servers. The advantages are as follows:
 - Multiple monitoring servers back up each other, improving reliability.
 - The load of a single server in monitoring BGP peers is reduced.
 - Different servers can be used to monitor routes from the same BGP peer in different address families, allowing each BGP service to be monitored by a different server.
- A monitoring server can monitor all BGP peers or a specified one.
- Configurations can be performed on the device to prevent the monitoring server from sending data (for example, by half-closing the TCP session or setting the window size to 0). Alternatively, the device can be configured to silently discard any data sent by the monitoring server.

Figure 1-363 Typical BMP networking



Benefits

BMP facilitates the monitoring of the BGP running status and reports security risks on networks in real time so that preventive measures can be taken promptly to improve network stability.

BGP Best External

Background

If multiple routes to the same destination are available, a BGP device selects one optimal route based on BGP route selection rules and advertises the route to its peers.

NOTE

For details about the BGP route selection rules, see *BGP Route Processing*.

However, in scenarios with primary and backup provider edges (PEs), if routes are selected based on the existing BGP route selection rules, no backup route may be available. As a result, route convergence takes a long time if a link fails. To address this problem, the BGP Best External feature was introduced.

Related Concepts

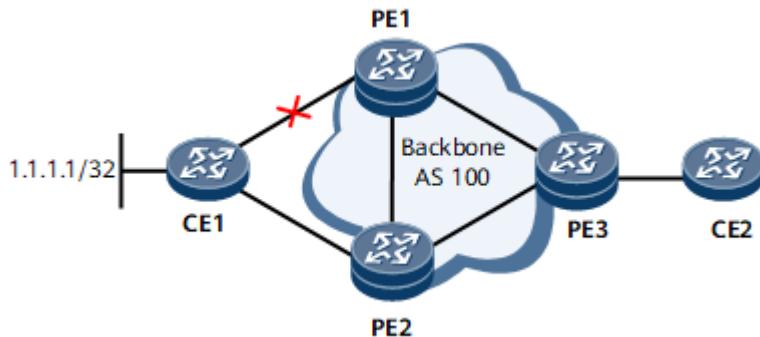
BGP Best External: After BGP Best External is enabled on a device, if the route preferentially selected by the device is an IBGP route, the device selects the sub-optimal route (EBGP or IBGP route) and advertises it to its peers. BGP Best External implements fast route convergence if the link between the device and a peer fails.

Best External route: The sub-optimal route selected after BGP Best External is enabled.

Networking with Master and Backup PEs

In the networking shown in [Figure 1-364](#), CE1 is dual-homed to PE1 and PE2. PE1 has a larger Local_Pref value than PE2. EBGP peer relationships are established between CE1 and PE1, and between CE1 and PE2. In addition, IBGP peer relationships are established among PE1, PE2, and PE3. PE1 and PE2 receive the same route to 1.1.1.1/32 from CE1, and PE2 receives the route from PE1. Of the two routes, PE2 preferentially selects the route from PE1 because PE1 has a larger Local_Pref value than CE1. PE2 does not advertise the route received from PE1 to PE3. Therefore, PE3 has only one route, which is advertised by PE1. If the primary link fails, traffic can be switched to the backup link only after routes are re-converged.

Figure 1-364 Networking with master and backup PEs



BGP Best External can be enabled on PE2 to address this problem. With BGP Best External, PE2 selects the EBGP route from CE1 and advertises it to PE3. In this case, a backup link is available. [Table 1-107](#) lists the differences with and without BGP Best External.

Table 1-107 Differences with and without BGP Best External

Feature Enabling Status	Route Advertisement	Optimal Route	Route Convergence in Case of a Link Fault
Before the BGP Best External feature is enabled	PE3 can receive only one route: CE1 → PE1 → PE3	CE1 -> PE1 -> PE3	The backup link can be selected only after PE1 and PE2 re-select a route.
After BGP Best External is enabled	PE3 receives two routes: CE1 → PE1 → PE3 CE1 → PE2 → PE3	CE1 -> PE1 -> PE3	Traffic can be directly switched to the backup link.

Usage Scenario

The master and backup provider edges (PEs) need to advertise sub-optimal routes (Best External routes) to peers to implement fast route convergence in the case of a link fault.

Benefits

As networks develop, services, such as Voice over Internet Protocol (VoIP), online video, and financial services, pose higher requirements for real-time transmission. After BGP Best External is deployed, if the optimal route selected by a device is an IBGP route, the device selects the suboptimal route and advertises it to BGP peers. This implements fast route convergence in the case of a link fault and reduces the impact of traffic interruption on services.

BGP Add-Path

Background

In a scenario with a route reflector (RR) and clients, if the RR has multiple routes to the same destination (with the same prefix), the RR selects an optimal route from these routes and then sends only the optimal route to its clients. Therefore, the clients have only one route to the destination. If a link along this route fails, route convergence takes a long time, which cannot meet the requirements on high reliability.

To address this issue, deploy the BGP Add-Path feature on the RR. With BGP Add-Path, the RR can send two or more routes with the same prefix to a specified peer. These routes can back up each other or load-balance traffic, which ensures high reliability in data transmission.

 NOTE

For details about BGP route selection and advertisement rules, see [BGP Route Processing](#).

Although BGP Add-Path can be deployed on any router, you are advised to deploy it on RRs.

The actual number of routes with the same prefix that the device can send to a peer is the smaller value between the configured maximum number of routes that BGP Add-Path allows the device to send to the peer and the number of existing Add-Path routes.

Add-Path takes effect only for routes learned from peers, not for local routes.

Related Concepts

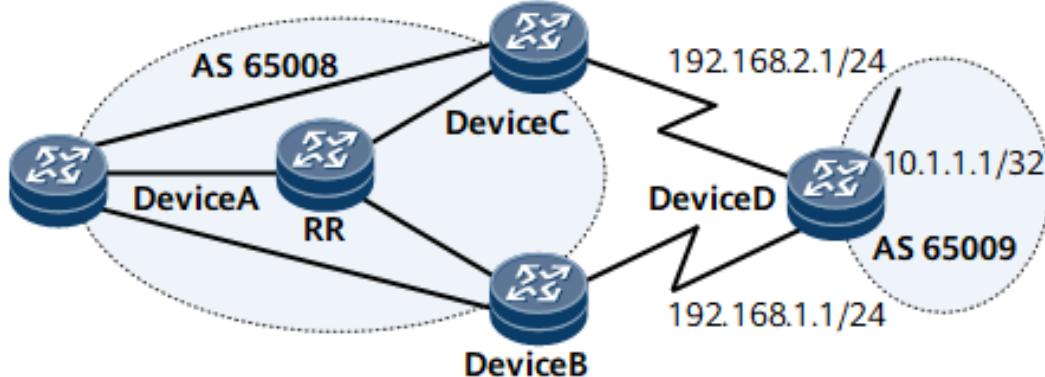
Add-Path routes: The routes selected by BGP after BGP Add-Path is configured.

Typical Networking

On the network shown in [Figure 1-365](#), DeviceA, DeviceB, and DeviceC are clients of the RR, and DeviceD is an EBGP peer of DeviceB and DeviceC. Both DeviceB and DeviceC receive a route to 10.1.1.1/32 from DeviceD.

DeviceB and DeviceC advertise the route 10.1.1.1/32 to the RR. The two routes have the same destination address but different next hops. The RR selects an optimal route based on BGP route selection rules and advertises the optimal route to DeviceA. Therefore, Device A has only one route to 10.1.1.1/32.

Figure 1-365 Networking with BGP Add-Path



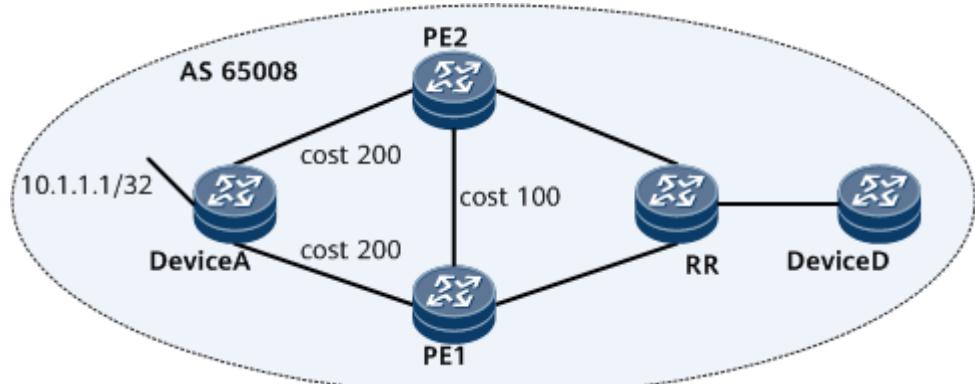
BGP Add-Path can be configured on the RR to control the maximum number of routes with the same prefix that the RR can send to DeviceA. Assume that the configured maximum number of routes with the same prefix that the RR can send to DeviceA is 2. [Table 1-108](#) lists the differences with and without BGP Add-Path.

Table 1-108 Differences with and without BGP Add-Path

Scenario	Route Advertisement	Route Convergence in Case of a Link Fault
Before BGP Add-Path is deployed	The RR advertises a route with 10.1.1.1/32 as the destination address and 192.168.1.1/24 as the next hop to DeviceA.	A new route must be selected to take over traffic after route convergence.
After BGP Add-Path is deployed	The RR advertises two routes destined for 10.1.1.1/32, one with next hop 192.168.1.1/24 and the other with next hop 192.168.2.1/24 to DeviceA.	When two links work in primary/backup mode and the primary link fails, traffic can be quickly switched to the backup link. When two links work in load-balancing mode and one of the links fails, all traffic on the faulty link is transferred to and transmitted over the other link.

Deploying BGP Add-Path may cause traffic loops. On the network shown in [Figure 1-366](#), PE1 and PE2 also function as RRs, and Add-Path is deployed on PE1, PE2, and RR. When reflecting routes to RR, PE1 and PE2 change the next hops of the routes to themselves. If the cost (cost1) of the path between PE1 and PE2 is 100, the cost (cost2) of the path between DeviceA and PE1 and the cost (cost3) of the path between DeviceA and PE2 are both 200, and the **network** command is run on DeviceA to import the local route 10.1.1.1 into BGP, a traffic loop may occur.

Figure 1-366 Networking with possible loops after BGP Add-Path is configured



The specific process is as follows:

1. DeviceA imports the local route 10.1.1.1 into BGP and advertises the route to its IBGP peers PE1 and PE2.

2. PE1 reflects the route 10.1.1.1 received from DeviceA to RR, which then reflects the route to PE2.
3. Because cost1 is smaller than cost3, PE2 preferentially selects the route with PE1 as the next hop. In addition, Add-Path is enabled between PE2 and RR. PE2 advertises the route with DeviceA as the next hop to RR, which then reflects the route to PE1.
4. Because cost1 is smaller than cost2, PE1 preferentially selects the route with PE2 as the next hop.
5. The traffic destined for 10.1.1.1 from DeviceD is looped between PE1 and PE2.

Usage Scenario

BGP Add-Path applies to scenarios in which an RR is deployed and needs to send multiple routes with the same prefix to clients to ensure data transmission reliability. BGP Add-Path supports route attribute-or prefix-based filtering to control Add-Path route advertisement in a refined manner.

BGP Add-Path is used in traffic optimization scenarios and allows multiple routes to be sent to the controller.

Benefits

Deploying BGP Add-Path can improve network reliability.

Route Dampening

Route instability is mainly reflected by route flapping. When a route flaps, it repeatedly disappears from the routing table and then reappears.

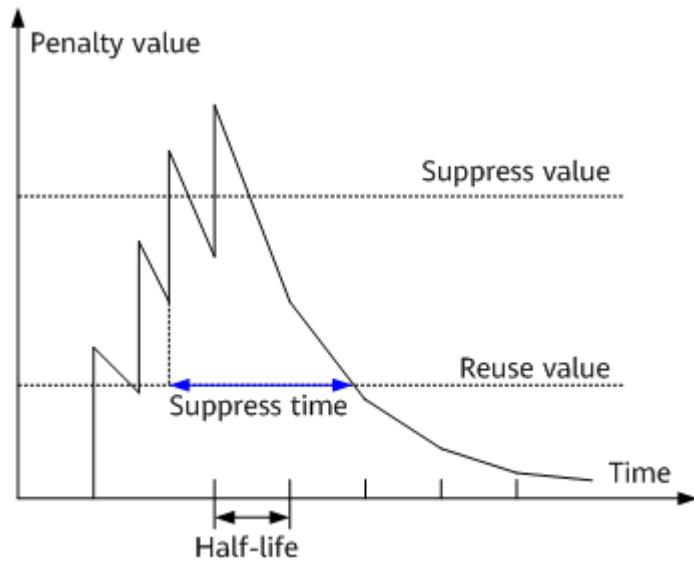
If route flapping occurs, the router sends an Update message to its peers. After the peers receive the Update message, they recalculate routes and update their routing tables. Frequent route flapping consumes bandwidth and CPU resources and even seriously affects network operations.

Route dampening can address this problem. In most cases, BGP is deployed on complex networks where routes change frequently. To reduce the impact of frequent route flapping, BGP adopts route dampening to suppress unstable routes.

BGP dampening measures route stability using a penalty value. The greater the penalty value, the less stable a route. Each time a route flaps, BGP increases the penalty value of the route. If the route changes from an active one to an inactive one, the penalty value increases by 1000. If the route is updated when it is active, the penalty value increases by 500. When the penalty value of a route exceeds the Suppress value, the route is suppressed. As a result, BGP does not add the route to the routing table or advertise any Update message to BGP peers.

The penalty value of a suppressed route reduces by half after a half-life period. When the penalty value decreases to the Reuse value, the route becomes reusable, and BGP adds the route to the IP routing table and advertises an Update packet carrying the route to BGP peers. The penalty value, suppression threshold, and half-life are configurable. [Figure 1-367](#) shows the process of BGP route dampening.

Figure 1-367 BGP route dampening



Suppression on BGP Peer Flapping

Suppression on BGP peer flapping is a way to suppress flapping. After this function is enabled, BGP peer relationships that flap continuously can be suppressed.

Background

BGP peer flapping occurs when BGP peer relationships are disconnected and then immediately re-established in a quick sequence that is repeated. Frequent BGP peer flapping is caused by various factors; for example, a link is unstable, or an interface that carries BGP services is unstable. After a BGP peer relationship is established, the local device and its BGP peer usually exchange all routes in their BGP routing tables with each other. If the BGP peer relationship is disconnected, the local device deletes all the routes learned from the BGP peer. Generally, a large number of BGP routes exist, and in this case, a large number of routes change and a large amount of data is processed when BGP peers frequently flap. As a result, a large number of resources are consumed, causing high CPU usage. To prevent this issue, a device supports suppression on BGP peer flapping. With this function enabled, the local device suppresses the establishment of the BGP peer relationship if it flaps continuously.

Related Concepts

ConnectFlaps: indicates the peer flapping count. The peer flapping count increases each time BGP peer flapping occurs. After flapping suppression exits, the statistics are cleared. Otherwise, the accumulated statistics are retained.

Peer flapping suppression period: The peer flapping suppression period is adjusted based on the **ConnectFlaps** value.

Idle hold timer: indicates the waiting period for re-establishing a connection with a peer. When the timer expires, BGP attempts to re-establish the connection with the BGP peer.

Half-life period: When the peer flapping counter (ConnectFlaps value) changes, the peer flapping count adjustment timer starts. When the timer expires (more than 1800s), the ConnectFlaps value is reduced by half. This period is called a half-life period.

Fundamentals

Entering flapping suppression

As shown in [Figure 1-368](#), when the ConnectFlaps value reaches a certain value (greater than 5), the Idle hold timer is used to suppress the establishment of the BGP peer relationship. The Idle hold timer value is calculated as follows:

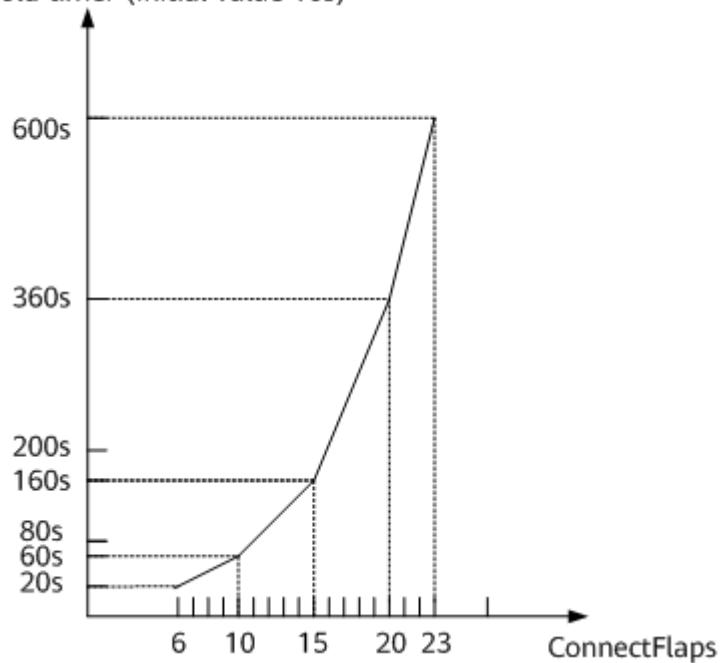
Idle hold timer = Initial waiting time + Peer flapping suppression period,

where, if the **peer timer connect-retry connect-retry-time** command is not run, the initial time that BGP waits to establish the peer relationship is 10s. If this command is run, the configured *connect-retry-time* value is used as the initial waiting time.

The peer flapping suppression period is processed as follows: If the ConnectFlaps value ranges from 1 to 5, the establishment of the peer relationship is not suppressed. If the ConnectFlaps value ranges from 6 to 10, the peer flapping suppression period increases by 10s each time the ConnectFlaps value is incremented by 1. If the ConnectFlaps value ranges from 11 to 15, the peer flapping suppression period increases by 20s each time the ConnectFlaps value is incremented by 1. For each of the following five-value ranges, the peer flapping suppression period increases by twice the time of the previous range each time the ConnectFlaps value is incremented by 1. The peer flapping suppression period no longer increases until the Idle hold timer reaches 600s. This prevents a BGP negotiation failure due to long-time suppression.

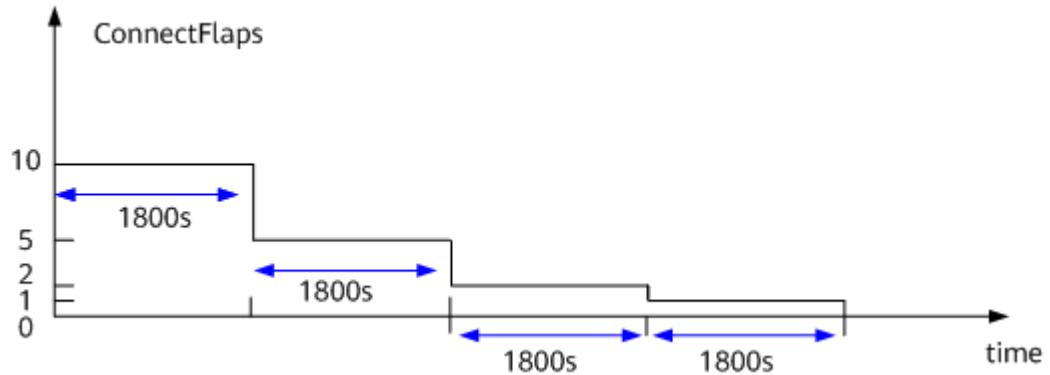
Figure 1-368 Relationship between the Idle hold timer and ConnectFlaps values when the initial waiting time is 10s

Idle hold timer (initial value 10s)



When the ConnectFlaps value changes, the peer flapping count adjustment timer starts. If the timer expires (more than 1800s have passed), the ConnectFlaps value is reduced by half, and a half-life period ends. In this case, if the ConnectFlaps value has not reached 0, the next half-life period will start. This process is cyclically repeated until the ConnectFlaps counter is reset. Assume that the ConnectFlaps value is 10. After four half-life periods elapse, the ConnectFlaps value changes to 0, as shown in [Figure 1-369](#).

Figure 1-369 Half-life periods



Exiting flapping suppression

Peer flapping suppression can be canceled in either of the following ways:

- Resetting the involved BGP process or BGP peer relationship
- Running a command that forcibly exits flapping suppression

BGP Recursion Suppression in Case of Next Hop Flapping

Background

In some scenarios, a large number of routes may recurse to the same next hop. If the next hop flaps due to a device fault, the system will become occupied processing reselection and re-advertisement of each involved route, leading to excessive resource consumption and high CPU usage. To address this problem, enable BGP recursion suppression in case of next hop flapping. If the next hop frequently flaps, recursion suppression in case of next hop flapping slows down route processing, conserving system resources and reducing CPU usage.

Fundamentals

After BGP recursion suppression in case of next hop flapping is enabled, a BGP device determines whether to increase, retain, or reduce the penalty value by comparing the flapping interval with the configured interval. When the penalty value exceeds 10, the BGP device suppresses route recursion to the corresponding next hop. For example, if the intervals for increasing, retaining, and clearing the penalty value are T1, T2, and T3, respectively, BGP calculates the penalty value as follows:

- Increases the penalty value by 1 if the flapping interval is less than T1.
- Retains the penalty value if the flapping interval is greater than or equal to T1, but less than T2.

- Reduces the penalty value by 1 if the flapping interval is greater than or equal to T2, but less than T3.
- Clears the penalty value if the flapping interval is greater than or equal to T3.

When the number of recursion suppressions in case of next hop flapping reaches a certain value (greater than 10), route processing is much lower than that without suppression.

Benefits

BGP recursion suppression in case of next hop flapping prevents the system from frequently processing reselection and re-advertisement of a large number of routes that are recursed to a flapping next hop, reducing system resource consumption and CPU usage.

BGP-LS

BGP-link state (LS) enables BGP to report topology information collected by IGPs to the upper-layer controller.

Background

BGP-LS is a new method of collecting topology information.

Without BGP-LS, the router uses an IGP (OSPF, OSPFv3, or IS-IS) to collect network topology information through routing information flooding and report the topology information of each area to the controller separately. This method has the following disadvantages:

- The controller must have high computing capabilities and support both the IGP and its algorithm.
- The controller cannot obtain complete information about the inter-IGP area topology. As a result, it cannot compute E2E optimal paths.
- The controller receives topology information from different routing protocols, making it difficult for the controller to analyze and process such information.

After BGP-LS is introduced, BGP summarizes topology information collected by IGPs and reports it to an upper-layer controller. With BGP's powerful route selection capabilities, BGP-LS has the following advantages:

- Lowers the requirements on the controller's computing and IGP capabilities.
- Facilitates path selection and computation on the controller by using BGP to summarize topology information in each process or AS and report the complete information directly to the controller.
- Requires only one routing protocol (BGP) to report information about the entire network's topology to the controller.

Related Concepts

BGP-LS provides a simple and efficient method of collecting topology information.

BGP-LS routes carry topology information and are classified into six types of routes that carry node, link, route prefix, IPv6 route prefix, SRv6 SID, and TE Policy information, respectively. These routes work together to transmit topology information.

BGP-LS Routes

Based on BGP, BGP-LS introduces a series of new Network Layer Reachability Information (NLRI) attributes to carry information about links, nodes, and IPv4/IPv6 prefixes. Such new NLRIs are called Link-State NLRIs. BGP-LS includes the MP_REACH_NLRI or MP_UNREACH_NLRI attribute in BGP Update messages to carry Link-State NLRIs.

BGP-LS defines the following types of Link-State NLRI:

- Node NLRI
- Link NLRI
- IPv4 Topology Prefix NLRI
- IPv6 Topology Prefix NLRI

In addition, the BGP-LS attribute is defined for Link-State NLRI to carry link, node, and IPv4/IPv6 prefix parameters and attributes. BGP-LS attributes are carried in TLVs and NLRIs in BGP-LS messages. These attributes, including the Node, Link, and Prefix attributes, are optional non-transitive BGP attributes.

Node NLRI format

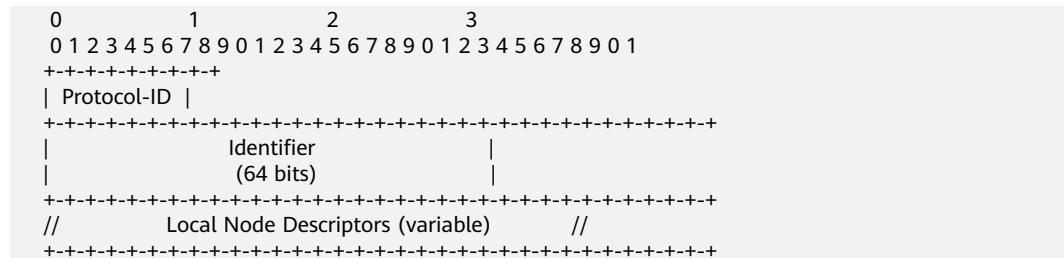
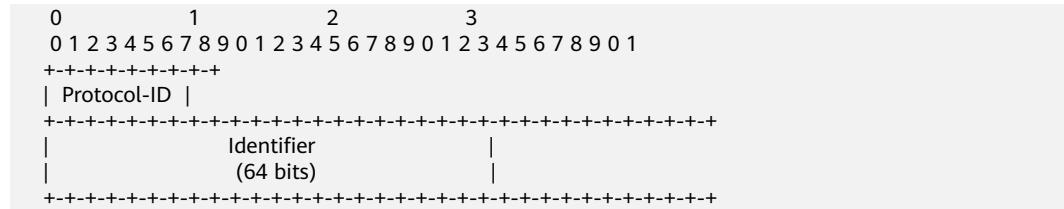


Table 1-109 Node NLRI field description

Field	Length	Description
Protocol-ID	1 octet	Protocol identifier, identifying a protocol such as IS-IS, OSPF, OSPFv3, or BGP.
Identifier	8 octets	Uniquely identifies a protocol instance when IS-IS, OSPFv3 multi-instance, or OSPF multi-instance is running.
Local Node Descriptors	Variable	The Local Node Descriptors TLV contains Node Descriptors for the local node of the link. This TLV consists of a series of Node Descriptor sub-TLVs.

Link NLRI format



```
//      Local Node Descriptors (variable)      //
+-----+-----+-----+-----+-----+-----+
//      Remote Node Descriptors (variable)      //
+-----+-----+-----+-----+-----+-----+
//      Link Descriptors (variable)      //
+-----+-----+-----+-----+-----+-----+
```

Table 1-110 Link NLRI field description

Field	Length	Description
Protocol-ID	1 octet	Protocol identifier, identifying a protocol such as IS-IS, OSPF, OSPFv3, or BGP.
Identifier	8 octets	Uniquely identifies a protocol instance when IS-IS, OSPFv3 multi-instance, or OSPF multi-instance is running.
Local Node Descriptors	Variable	The Local Node Descriptors TLV contains Node Descriptors for the local node of the link. This TLV consists of a series of Node Descriptor sub-TLVs.
Remote Node Descriptors	Variable	The Remote Node Descriptors TLV contains Node Descriptors for the remote node of the link.
Link Descriptors	Variable	The Link Descriptors field is a set of TLV triplets. This field uniquely identifies a link among multiple parallel links between a pair of devices.

IPv4/IPv6 Topology Prefix NLRI

```
0   1   2   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+
| Protocol-ID |
+-----+-----+-----+
| Identifier   |
| (64 bits)    |
+-----+-----+-----+
//  Local Node Descriptors (variable)  //
+-----+-----+-----+
//  Prefix Descriptors (variable)    //
+-----+-----+-----+
```

Table 1-111 Link NLRI field description

Field	Length	Description
Protocol-ID	1 octet	Protocol identifier, identifying a protocol such as IS-IS, OSPF, OSPFv3, or BGP.
Identifier	8 octets	Uniquely identifies a protocol instance when IS-IS, OSPFv3 multi-instance, or OSPF multi-instance is running.

Field	Length	Description
Local Node Descriptors	Variable	The Local Node Descriptors TLV contains Node Descriptors for the local node of the link. This TLV consists of a series of Node Descriptor sub-TLVs.
Prefix Descriptors	Variable	The Prefix Descriptors field is a set of TLV triplets. This field uniquely identifies a prefix originated by a node.

BGP-LS Route Formats

Format of node routes

For example, a node route is in the format of [NODE][ISIS-LEVEL-1][IDENTIFERO][LOCAL[as100][bgp-ls-identifier10.1.1.2][ospf-area-id0.0.0.0][igp-router-id0000.0000.0001.00]].

Node routes carry node information.

[Table 1-112](#) describes the fields in node routes.

Table 1-112 Description of the fields in node routes

Item	Description
NODE	Field indicating that the BGP-LS route is a node route.
ISIS-LEVEL-1	Protocol that collects topology information. The protocol is IS-IS in this example.
IDENTIFERO	BGP-LS identifier of the protocol that collects topology information.
LOCAL	Field indicating information of a local node.
as	BGP-LS domain AS number.
bgp-ls-identifier	BGP-LS domain ID.
ospf-area-id	OSPF area ID.
igp-router-id	IGP router ID, generated by the IGP that collects topology information. The router ID is obtained from the NET of an IS-IS process in this example.

Format of link routes

For example, a link route is in the format of [LINK][ISIS-LEVEL-1][IDENTIFERO][LOCAL[as255.255][bgp-ls-identifier192.168.102.4][ospf-area-id0.0.0.0][igp-

router-id0000.0000.0002.01]][REMOTE[as255.255][bgp-ls-identifier192.168.102.4]
[ospf-area-id0.0.0.0][igp-router-id0000.0000.0002.00]][LINK[if-address0.0.0.0]
[peer-address0.0.0.0][if-address::][peer-address::][mt-id0]].

Link routes carry information about links between devices.

Table 1-113 describes the fields in link routes.

Table 1-113 Description of the fields in link routes

Item	Description
LINK	Field indicating that the BGP-LS route is a link route.
ISIS-LEVEL-1	Protocol that collects topology information. The protocol is IS-IS in this example.
IDENTIFERO	BGP-LS identifier of the protocol that collects topology information.
LOCAL	Field indicating information of a local node.
as	BGP-LS domain AS number.
bgp-ls-identifier	BGP-LS domain ID.
ospf-area-id	OSPF area ID.
igp-router-id	IGP router ID, generated by the IGP that collects topology information. The router ID is obtained from the NET of an IS-IS process in this example.
REMOTE	Field indicating information of a remote node.
if-address	IP address of the local interface.
peer-address	IP address of the remote interface.
mt-id	ID of the topology to which an IGP interface is bound.

Format of prefix routes

For example, a prefix route is in the format of [IPV4-PREFIX][ISIS-LEVEL-1]
[IDENTIFERO][LOCAL[as100][bgp-ls-identifier192.168.102.3][ospf-area-id0.0.0.0]
[igp-router-id0000.0000.0001.00]][PREFIX[mt-id0][ospf-route-type0]
[prefix192.168.102.0/24]].

Prefix routes carry information about reachable network segments.

Table 1-114 describes the fields in prefix routes.

Table 1-114 Description of the fields in prefix routes

Item	Description
IPV4-PREFIX	Field that indicates an IPv4 prefix route. Prefix routes are classified as IPv4 prefix routes or IPv6 prefix routes. The router cannot generate IPv6 prefix routes, but it can process the IPv6 prefix routes received from non-Huawei devices.
ISIS-LEVEL-1	Protocol that collects topology information. The protocol is IS-IS in this example.
IDENTIFERO	BGP-LS identifier of the protocol that collects topology information.
LOCAL	Field indicating information of a local node.
as	BGP-LS domain AS number.
bgp-ls-identifier	BGP-LS domain ID.
ospf-area-id	OSPF area ID.
igp-router-id	IGP router ID, generated by the IGP that collects topology information. The router ID is obtained from the NET of an IS-IS process in this example.
PREFIX	Field indicating an IGP route.
mt-id	ID of the topology to which an IGP interface is bound.
ospf-route-type	OSPF route type: <ul style="list-style-type: none"> ● 1: Intra-Area ● 2: Inter-Area ● 3: External 1 ● 4: External 2 ● 5: NSSA 1 ● 6: NSSA 2
prefix	Prefix of an IGP route.

Format of TE Policy routes

For example: [TEPOLICY][SEGMENT-ROUTING][IDENTIFERO][LOCAL[as100][bgp-ls-identifier1.1.1.1][bgp-router-id1.1.1.2][ipv4-router-id1.1.1.9][ipv6-router-id2001:DB8:1::1]][TE[protocol-origin3][Flag0][endpoint2.2.2.2][color123][originator-as0][originator-address0.0.0.0][discriminator500]]

The routes carry information about SR TE Policy-related topology and status.

Table 1-115 describes the fields in TE Policy routes.

Table 1-115 Description of the fields in TE Policy routes

Item	Description
TEPOLICY	Field indicating that the BGP-LS route is a TE Policy route.
SEGMENT-ROUTING	Segment routing.
IDENTIFIER0	BGP-LS identifier of the protocol that collects topology information.
LOCAL	Field indicating information of a local node.
as	BGP-LS domain AS number.
bgp-ls-identifier	BGP-LS domain ID.
bgp-router-id	BGP router ID.
ipv4-router-id	IPv4 router ID.
ipv6-router-id	IPv6 router ID.
TE	Traffic engineering.
protocol-origin2	Protocol origin of the primary path over an SR-MPLS TE Policy tunnel.
Flag	Flag bit.
endpoint	Destination IP address of an SR-MPLS TE Policy tunnel.
color	Color attribute carried in SR-MPLS TE Policy routes.
originator-as	Originator AS number configured for the primary path over an SR-MPLS TE Policy tunnel.
originator-address	Originator address configured for the primary path over an SR-MPLS TE Policy tunnel.
discriminator	Discriminator of the primary path over an SR-MPLS TE Policy tunnel.

Format of IPv6 prefix routes

For example: [IPV6-PREFIX][ISIS-LEVEL-2][IDENTIFIER100][LOCAL[as200][bgp-ls-identifier192.168.11.11][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]] [PREFIX[mt-id0][ospf-route-type0][prefix2001:DB8:1::1/64]]

The routes carry information about reachable network segments.

Table 1-116 describes the fields in IPv6 prefix routes.

Table 1-116 Description of the fields in IPv6 prefix routes

Item	Description
IPV6-PREFIX	Field that indicates an IPv6 prefix route. Prefix routes are classified as IPv4 prefix routes or IPv6 prefix routes. The router cannot generate IPv6 prefix routes, but it can process the IPv6 prefix routes received from non-Huawei devices.
ISIS-LEVEL-2	Protocol that collects topology information. The protocol is IS-IS in this example.
IDENTIFIER	BGP-LS identifier of the protocol that collects topology information.
LOCAL	Field indicating information of a local node.
as	BGP-LS domain AS number.
bgp-ls-identifier	BGP-LS domain ID.
ospf-area-id	OSPF area ID.
igp-router-id	IGP router ID, generated by the IGP that collects topology information. The router ID is obtained from the NET of an IS-IS process in this example.
PREFIX	Field indicating an IGP route.
mt-id	ID of the topology to which an IGP interface is bound.
ospf-route-type	OSPF route type: <ul style="list-style-type: none"> ● 1: Intra-Area ● 2: Inter-Area ● 3: External 1 ● 4: External 2 ● 5: NSSA 1 ● 6: NSSA 2
prefix	Prefix of an IGP route.

Format of SRv6 SID routes

For example, an SRv6 SID route is in the format of [SRV6-SID][ISIS-LEVEL-2][IDENTIFIER100][LOCAL[as200][bgp-ls-identifier192.168.11.11][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]][SID[mt-id0][sid2001:db8:1::1]].

Such routes carry information about reachable network segments.

Table 1-117 describes the fields in this type of route.

Table 1-117 Description of the fields in SRv6 SID routes

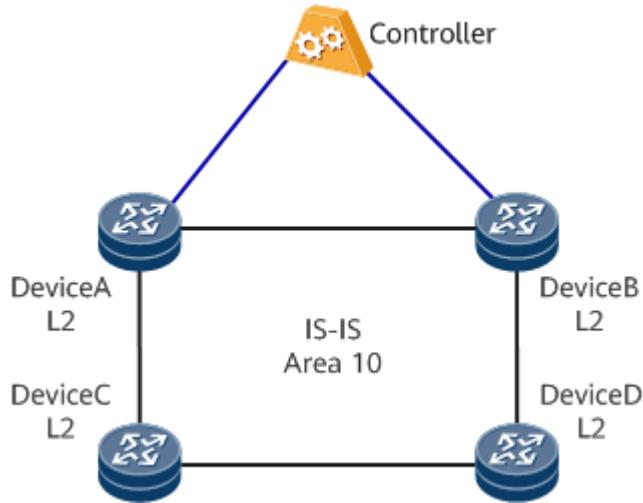
Item	Description
SRV6-SID	Field indicating an SRv6 SID route.
ISIS-LEVEL-2	Protocol that collects topology information. The protocol is IS-IS in this example.
IDENTIFIER	BGP-LS identifier of the protocol that collects topology information.
LOCAL	Field indicating information of a local node.
as	BGP-LS domain AS number.
bgp-ls-identifier	BGP-LS domain ID.
ospf-area-id	OSPF area ID.
igp-router-id	IGP router ID, generated by the IGP that collects topology information. The router ID is obtained from the NET of an IS-IS process in this example.
mt-id	ID of the topology to which an IGP interface is bound.
sid	SRv6 SID value.

Typical Networking

Collecting topology information in an IGP area

In [Figure 1-370](#), DeviceA, DeviceB, DeviceC, and DeviceD use IS-IS to communicate with each other at the IP network layer. DeviceA, DeviceB, DeviceC, and DeviceD are all Level-2 devices in the same area (area 10). After BGP-LS is deployed on any one of the devices (DeviceA, DeviceB, DeviceC, and DeviceD) and this device establishes a BGP-LS peer relationship with the controller, topology information of the entire network can be collected and reported to the controller. Reliability can be improved by deploying BGP-LS on two or more devices and establishing a BGP-LS peer relationship between each BGP-LS device and the controller. Because the BGP-LS devices collect the same topology information, they back up each other. This means that the topology information can be reported promptly if any of the BGP-LS devices fails.

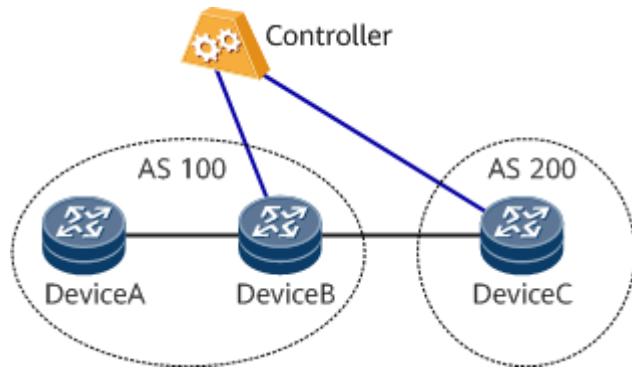
Figure 1-370 Networking in which topology information is collected within an IGP area



Collecting BGP inter-AS topology information

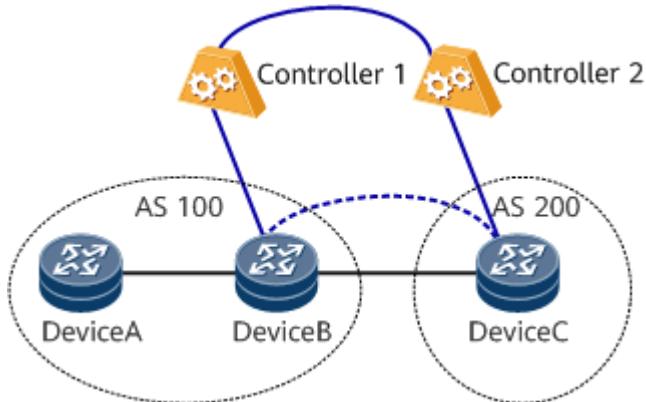
In [Figure 1-371](#), DeviceA and DeviceB belong to the same AS, and an IS-IS neighbor relationship is established between them. BGP is not enabled on DeviceA in the AS. An EBGP peer relationship is established between DeviceB and DeviceC. If BGP-LS is not enabled, topology information cannot be transmitted between the ASs. Because the devices collect information about only their own AS, the topology information in AS 100 is different from that in AS 200. In this case, BGP-LS must be enabled on at least one device in each AS and this device must establish a BGP-LS peer relationship with the controller. To ensure that topology information can be collected and reported reliably, enable that two or more devices in each AS are connected to the controller.

Figure 1-371 Networking 1 in which topology information is collected across BGP ASs



On the network shown in [Figure 1-372](#), two controllers are each connected to a device in a different AS. If both controllers need to obtain information about the entire network's topology, a BGP-LS peer relationship needs to be established between the controllers or between the devices (DeviceB and DeviceC in this example) connected to the controllers.

Figure 1-372 Networking 2 in which topology information is collected across BGP ASs



NOTE

To minimize the number of connections with the controllers, one or more devices can be used as BGP-LS RRs, which then function as proxies to establish BGP-LS peer relationships between the devices and controllers.

Usage Scenario

The router functions as a forwarder and reports topology information to the controller for topology monitoring and traffic control.

Benefits

BGP-LS offers the following benefits:

- Reduces computing capability requirements on the controller.
- Allows the controller to gain the complete inter-AS topology information.
- Requires only one routing protocol (BGP) to report topology information to the controller.

BGP RPD

Background

Route policy distribution (RPD) is used to distribute route-policies dynamically.

Without RPD, route-policies can be generated only through manual configuration, and then the route-policies are applied to specified peers. Such a generation mode is not applicable when the route-policies need to be adjusted dynamically and frequently, for example, in the inbound traffic optimization scenario where the controller monitors the traffic bandwidth usage on the network in real time, and users perform traffic optimization based on the analysis result. In this case, route-policies need to be used to control the route selection on the peer end by modifying attributes of specified routes. However, the traffic bandwidth usage constantly changes, leading to the constant changes of traffic optimization policies. In this case, manual delivery of route-policies is not suitable. RPD provides a dynamic route-policy distribution mode for the controller. With RPD, route-policy information is transmitted through the RPD address family.

After RPD is configured, the controller monitors and controls traffic in real time. The overall configurations are performed on the controller, not on the local device, which functions as a forwarder and only receives and executes policies delivered by controller.

Related Concepts

RPD route: Carries route-policy information and distributes the information to peers in the BGP RPD address family. After learning the RPD route, the receiver converts it into a route-policy and applies the policy.

RPD Route Format

RPD routes are in the format of policy type (export policy)/peer address/policy ID, for example, 1/1.1.1.1/1.

Table 1-118 describes the fields in the routes.

Table 1-118 RPD route description

Field	Description
Policy type	Specifies the policy type. Currently, only export policies are supported.
Peer address	Specifies the peer address used by the policy.
Policy ID	Specifies the ID of a policy.

Route-policies carried in RPD routes are encapsulated through the WideCommunity attribute. [Figure 1-373](#) shows the WideCommunity format used by RPD routes.

Figure 1-373 WideCommunity format used by RPD routes

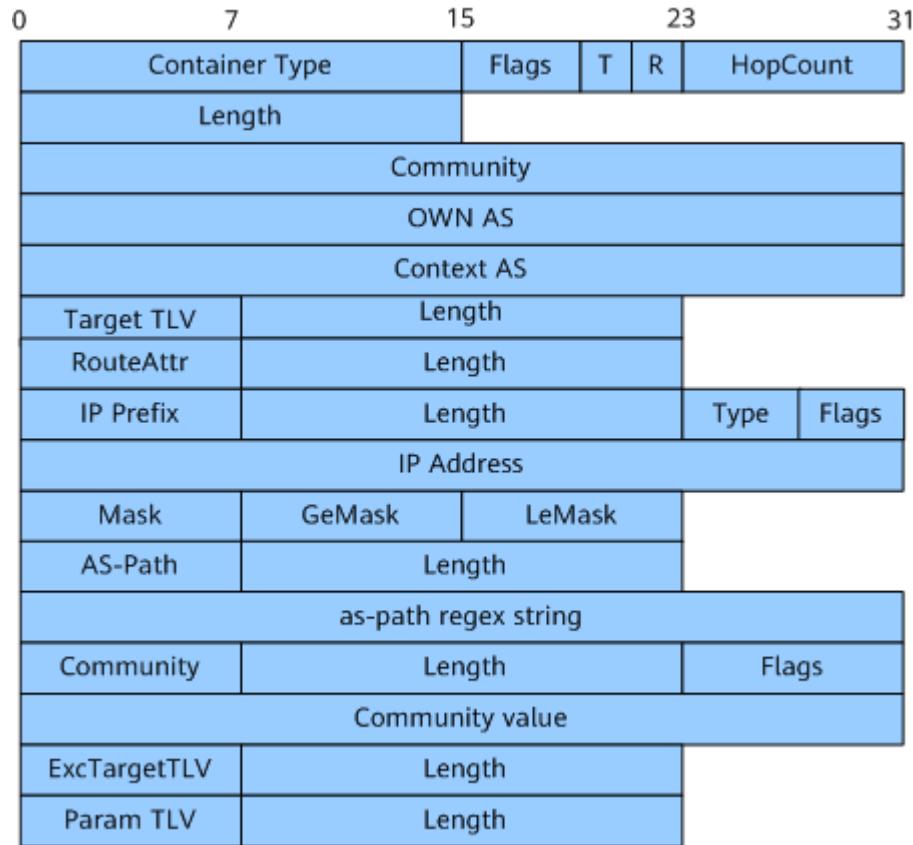


Table 1-119 Description of fields in the WideCommunity format used by RPD routes

Field	Length (in bits)	Description
Container Type	16	The value is 1, indicating the WideCommunity attribute.
Flags	8	The R bit is 0, indicating a private attribute. The T bit is 0, which is not used currently.
HopCount	8	This field is not used currently. The value is 1.
Length	16	Total packet length.
Community	32	Value of WideCommunity. Each value identifies a specific function of WideCommunity. If the value is MATCH AND SET ATTR , the attributes of the routes that match specific conditions are modified. If the value is MATCH AND NOT ADVERTISE , the routes that match specific conditions are not advertised.

Field	Length (in bits)	Description
OWN AS	32	AS of the controller.
Context AS	32	AS number of the local device (forwarder).
Target TLV	8	Target TLV.
Length	16	Target TLV length.
RouteAttr	8	Route attribute type. The TLV has three sub-TLVs: IP Prefix, AS-Path, and Community.
Length	16	Length of the route attribute type.
IP Prefix	8	IP address prefix.
Length	16	Length of IP address prefix.
Type	4	Matching type of the IP address prefix. <ul style="list-style-type: none"> • 0: exact matching • 1: matches the routes that carry the prefix and a mask whose length is greater than or equal to the specified mask length. • 2: matches the routes that carry the prefix and a mask whose length is less than or equal to the specified mask length. • 3: matches the routes that carry the prefix and a mask whose length is within the specified range.
Flags	4	This field is not used currently.
IP Address	32	Peer IP address
Mask	8	Mask of the IP address.
GeMask	8	Used to specify a range. The value of this field must be less than or equal to the length of the Mask or be 0.
LeMask	8	Used to specify a range. The value of this field must be greater than the length of the Mask or be 0.
AS-Path	8	AS_Path.
Length	16	AS_Path length.

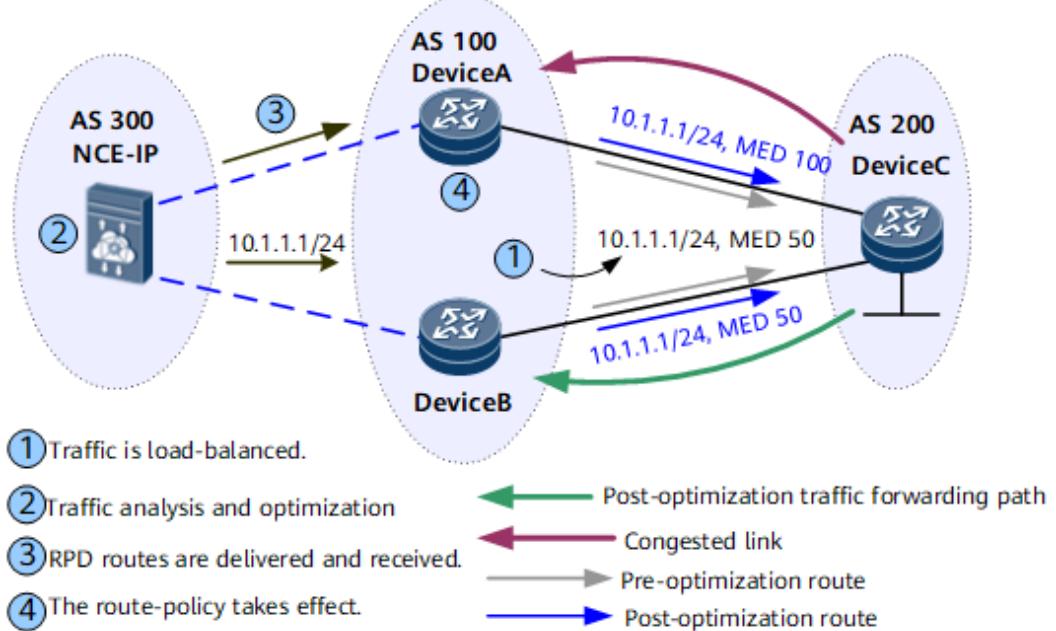
Field	Length (in bits)	Description
as-path regex string	32	Content of the AS_Path, which is presented using a regular expression. The maximum length of the AS_Path is 1024 bytes.
Community	8	Community attribute.
Length	16	Length of the community attribute.
Flags	8	This field is not used currently.
Community value	32	Community attribute value.
ExcTargetTLV	8	This field is not used currently. Even if this TLV has a value, it is also ignored.
Length	16	ExcTargetTLV length
Param TLV	8	Content of the action to be performed. The format depends on the Community value, and the TLV content also varies according to the Community value. If the Community value is MATCH AND SET ATTR , the MED, community, or AS_Path attribute is set. If the Community value is MATCH AND NOT ADVERTISE , the Param TLV is empty, without any sub-TLV.
Length	16	Param TLV length

Implementation

In the following example, the typical networking of inbound traffic optimization is used to describe how RPD works to implement traffic optimization:

Figure 1-374 shows an inbound traffic optimization scenario. The controller collects traffic information from forwarders and performs analysis and calculation to identify the routes to be adjusted. After a traffic optimization policy is configured, the controller converts the policy into an RPD route and delivers the route to the forwarders.

Figure 1-374 Typical networking of inbound traffic optimization



In **Figure 1-374**, the implementation of inbound traffic optimization is as follows:

1. Before traffic optimization is implemented, the MED values of the BGP routes advertised from DeviceA to DeviceC and from DeviceB to DeviceC are both 50, and traffic is load-balanced.
2. The controller collects traffic information from forwarders, performs calculation, and finds that the path from DeviceC to DeviceA is congested. In this case, it is expected that the traffic that is from AS 200 and destined for 10.1.1.1/24 enters AS 100 through DeviceB rather than DeviceA. After a traffic optimization policy is configured on the controller, the controller converts it into an RPD route and delivers the route to DeviceA to instruct DeviceA to increase the MED value of the route to be advertised to AS 200 to 100. The MED value of the route advertised by DeviceB to AS 200 remains unchanged.
3. After receiving the RPD route delivered by the controller, DeviceA executes the route-policy carried in the RPD route.
4. The RPD route-policy takes effect. After receiving the routes destined for 10.1.1.1/24 from DeviceA and DeviceB, DeviceC in AS 200 selects the route received from DeviceB because its MED value is smaller than the MED value of the route received from DeviceA. In this case, traffic that is from AS 200 and destined for 10.1.1.1/24 is forwarded through DeviceB.

NOTE

In this scenario, forwarders receive the policies delivered by the controller and adjust route attributes (MED, community, or AS_Path) based on the policies. The forwarders follow the policies strictly when advertising routes, but are not responsible for the traffic optimization results. You can check the traffic optimization results in real time through the controller.

Usage Scenario

The IP network optimization solution provides users with a method of on-demand traffic scheduling to make full use of network bandwidth. In the IP network

optimization solution, the inbound traffic optimization solution is provided to meet requirements of inbound traffic optimization in MAN ingress or IGW scenarios. In this solution, the routers function as forwarders and need to be configured with the RPD feature so that they execute the route-policies carried in the RPD routes delivered by the controller to dynamically adjust traffic for inbound traffic optimization.

Benefits

In traffic optimization scenarios, this feature spares manual BGP route-policy maintenance, which is complex, time-consuming, and error-prone. Therefore, this feature reduces maintenance workload and improves maintenance quality.

BGP Multi-instance

Background

By default, all BGP routes are stored in the BGP basic instance, and separate route management and maintenance are impossible. To address this problem, BGP multi-instance is introduced. A device can simultaneously run two BGP instances: a BGP basic instance and a BGP multi-instance. The two BGP instances are independent of each other and can have either the same AS number or different AS numbers. BGP multi-instance can achieve separate route management and maintenance by having different address families deployed in the BGP basic instance and BGP multi-instance.

Basic Concepts

A BGP instance can be classified as either of the following:

- BGP basic instance (BGP view), such as **bgp 100**
- BGP multi-instance (BGP multi-instance view), such as **bgp 100 instance a**

A device can run the BGP basic instance and BGP multi-instance simultaneously. Their AS numbers can be either the same or different. The BGP multi-instance process functions in a similar way to the BGP basic instance process.

Implementation

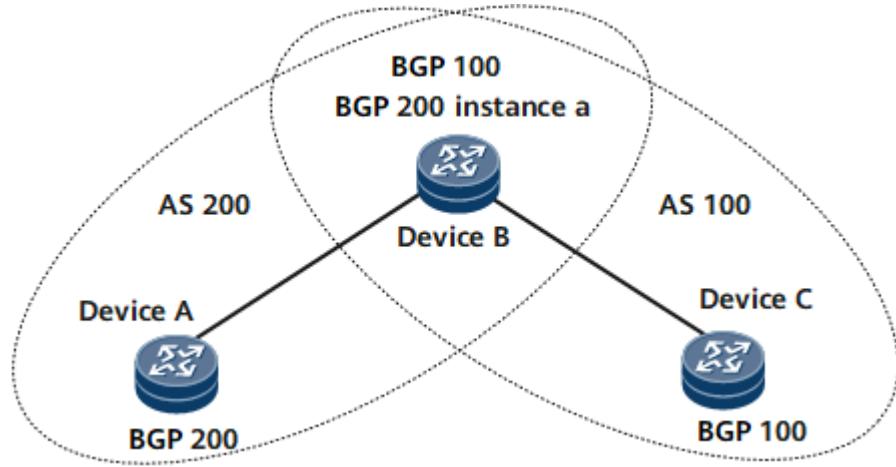
On the network shown in [Figure 1-375](#), to isolate private and public network services, specifically, to deploy public network services between Device A and Device B and private network services between Device B and Device C, configure BGP as follows:

- Configure BGP basic instance **bgp 200** on Device A.
- Configure BGP basic instance **bgp 100** and BGP multi-instance **bgp 200 instance a** on Device B.
- Configure BGP basic instance **bgp 100** on Device C.

The public network BGP-IPv4 unicast address family is enabled in BGP basic instances on Device A and Device B and a public network EBGP peer relationship is established for the exchange of public network routes. The VPN address family is enabled in the BGP multi-instance on Device B and BGP basic instance on Device C, and an EBGP-VPN peer relationship is established for the exchange of VPN routes. Check route information on Device A, Device B, and Device C. If

Device A has only public network routes, Device B has both VPN and public network routes, and Device C has only VPN routes, instance-specific management and maintenance of routes can be achieved.

Figure 1-375 BGP multi-instance for service isolation



BGP SR LSP

BGP Segment Routing (SR) uses BGP as the control-plane protocol to transmit SR information. BGP uses the prefix SID attribute to advertise Segment Routing global block (SRGB) and Label-Index information for unified SR label deployment, after which BGP SR LSPs can be established. BGP SR LSPs are essentially BGP LSPs and are created in a similar way. The data forwarding process using BGP SR LSPs is also similar to that using BGP LSPs. The main difference between BGP SR LSPs and BGP LSPs lies in the label distribution mode. BGP SR LSPs use the SR label distribution mode, in which a label value is allocated to a specified route in a fixed mode (Label-Index+SRGB). This mode is a static label configuration mode, whereas BGP LSPs use a dynamic label allocation mode.

BGP SR LSP Creation

BGP SR LSPs are created primarily based on prefix SIDs. The destination node uses BGP to advertise a prefix SID, creates an LSP, and delivers the forwarding entry to guide data packet forwarding. Each forwarder parses the prefix SID and obtains the outgoing label and outbound interface based on the tunnel forwarding table to guide traffic forwarding. On the network shown in [Figure 1-376](#), each pair of neighboring nodes — A and B, B and C, and C and D — establish an LDP LSP, and belong to different IGP areas. To ensure service interworking between node A and node D, an E2E BGP SR LSP needs to be established. This section describes only the process of establishing a BGP SR LSP. For details about the process of establishing an LDP LSP, see LDP LSP Establishment.

Figure 1-376 Prefix SID-based BGP SR LSP creation

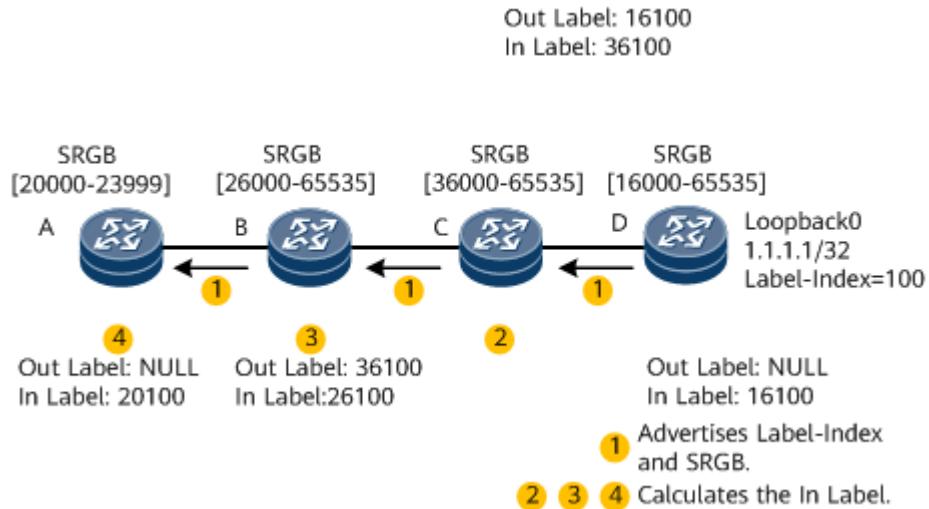


Table 1-120 describes the process of establishing a prefix SID-based BGP SR LSP.

Table 1-120 Process of creating a BGP SR LSP

Step	Device	Operation
1	D	An SRGB and Label-Index are configured on node D. The incoming label (In Label for short) of the route to 1.1.1.1/32 is 16100 on node D. In this case, node D instructs node C to use 16100 as the BGP SR LSP label for the route to 1.1.1.1/32. Node D creates an ILM entry to guide the processing of the In Label, encapsulates its SRGB and Label-Index into the Prefix SID attribute of a BGP route, and advertises the BGP route to its BGP peers.
2	C	After parsing the BGP message advertised by node D, node C sets the outgoing label (Out Label for short) to the In Label advertised by node D, instructs the tunnel management module to update the BGP LSP information, and delivers an NHLFE entry. In addition, node C calculates the In Label by adding the start value of its SRGB [36000-65535] and the Label-Index carried in the received message. The calculated In Label is 36100 (36000 + 100). After applying for the label, node C creates an ILM entry.
3	B	The process is similar to that on node C. The Out Label is 36100, and the In Label is 26100 (26000 + 100).
4	A	The process is similar to that on node B, and the In Label is 20100.

Data Forwarding

BGP SR LSPs use the same three types of label operations as those used in MPLS: push, swap, and pop.

Figure 1-377 Prefix SID-based data forwarding

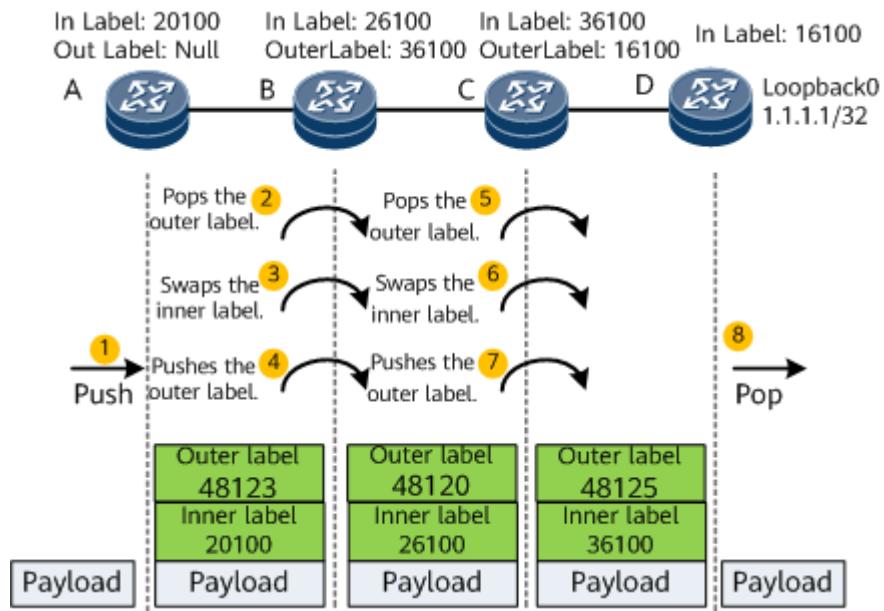


Table 1-121 describes the process of data forwarding on the network shown in **Figure 1-377**.

Table 1-121 Data forwarding process

Step	Device	Operation
1	A	<p>Node A receives a data packet and finds that the destination IP address of the packet is 1.1.1.1. Node A searches for the corresponding BGP LSP and pushes an inner MPLS label 20100 to the packet.</p> <p>Node A searches the inner and outer label mapping table, pushes an outer MPLS label 48123 to the packet, and forwards the packet through the outbound interface.</p> <p>NOTE</p> <p>To implement MPLS forwarding, each node creates an inner and outer label mapping table. Take node A as an example. According to the BGP SR LSP, when the inner label of a packet is 20100, the destination address is 1.1.1.1. According to the LDP LSP, when the packet needs to be sent to node B, the outer label 48123 needs to be added to the packet. Therefore, when the inner label of a packet is 20100, the outer label 48123 needs to be added. This mapping is recorded in the inner and outer label mapping table. With this table, node A does not need to query the IP routing table for an entry to send the packet to node B. Instead, node A only needs to query its inner and outer label mapping table for packet forwarding.</p>

Step	Device	Operation
2	B	After receiving the labeled packet, node B searches for an LDP LSP and pops the outer label 48123. Then, node B searches for a BGP LSP and swaps the inner label from 20100 to 26100. Finally, node B queries its inner and outer label mapping table, pushes an outer label 48120 to the packet, and forwards the packet through the outbound interface.
3	C	The operations on node C are similar to those on node B.
4	D	After receiving the labeled packet, node D searches for an LDP LSP and pops the outer label 48125 from the packet. Then, node D searches for a BGP LSP, pops the inner label 36100 from the packet, and forwards the packet to the destination.

BGP SAVNET

BGP source address validation network (SAVNET) implements distributed source address validation through BGP.

Context

Source address validation (SAV) is an important method to eliminate the source address forgery attack, which is one of the most concerned network security threats currently. SAV establishes a mapping between a source address and an inbound interface of a router and checks whether messages from the source address reach the router through this inbound interface. However, existing solutions (such as URPF) have limitations in accuracy, flexibility, and deployability, which limits the effect of defending against source address forgery attacks in practice.

To generate accurate SAV rules, a router deployed with SAV needs to know exactly which interface or interfaces a message carrying a valid source address should arrive at. The BGP SAVNET address family enables all nodes on the entire network to obtain the valid source prefix list of each node through source prefix advertisement (SPA). Then, the BGP SAVNET address family obtains the valid inbound interfaces of valid source prefixes through destination prefix probing (DPP) and generates SAV rules.

After SAVNET is configured on all devices in an AS, the mapping between the source address and the valid inbound interface of data messages can be generated on each node through SPA and DPP to filter out source address forgery attack traffic and allow valid traffic to pass through.

NOTE

BGP SAVNET is not supported on the NetEngine 8100 X8.

Related Concepts

SAV: in this feature refers to validation on the source address carried in each data message.

SPA: refers to the process in which a local node obtains a list of locally originated valid source prefixes through import of local routes or manual configuration and advertises the list to other nodes.

Multi-homed ingress interface group (MIIG): All inbound interfaces that connect the same subnet to the SAVNET deployment domain must be identified as one MIIG.

SPA message: refers to a data message that carries the SPA function and is encapsulated as a BGP Update message at the outer layer. A single SPA route is a BGP NLRI.

DPP: refers to the process in which a probing origin node sends a probing message to simulate a real path of valid data traffic and each relay node through which the message passes records source prefix information of the origin node and the valid inbound interface of the probing message to generate SAV rules.

DPP message: refers to a data message that carries the DPP function and is encapsulated as a BGP Refresh message at the outer layer.

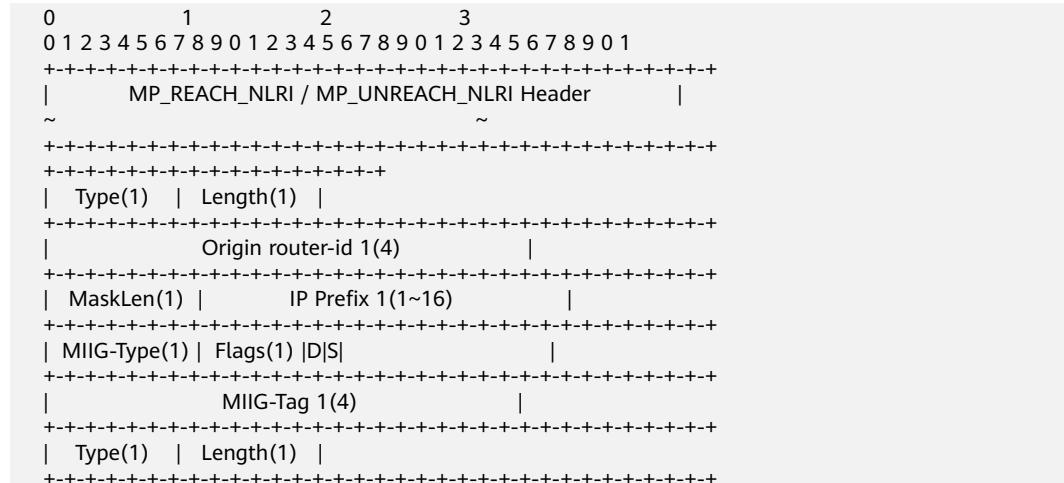
SAV rule: refers to the mapping (generated through DPP) between the source address of a data message and the valid inbound interface. A list of such rules is called a SAV-rule-table.

SAV in the forwarding plane: refers to the process in which the system validates service messages based on SAV rules by matching their source prefixes and valid inbound interfaces to filter out invalid service traffic.

SPA Message

An SPA message is a data message that carries the SPA function and uses the MP_REACH_NLRI and MP_UNREACH_NLRI attributes as containers of SPA routes (SPA routes are carried in BGP Update messages as the MP_REACH_NLRI or MP_UNREACH_NLRI attribute). The format of an SPA message is as follows:

SPA message format



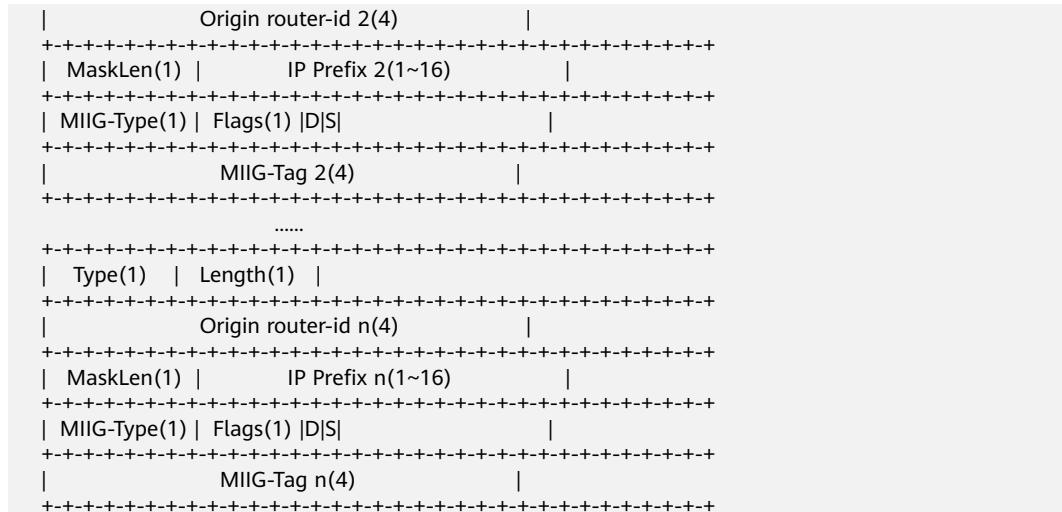


Table 1-122 Description of fields in the MP_REACH_NLRI format used by SPA routes

Field	Length (in Bits)	Description
MP_REACH_NLRI	Variable	Multiprotocol reachable NLRI.
MP_UNREACH_NLRI	Variable	Multiprotocol unreachable NLRI.
Type	8	SPA source prefix type. The current value is 1.
Length	8	Length of an SPA route NLRI.
Origin router-id	32	ID of the origin node, uniquely identifying a network device in an AS.
MaskLen	8	IP prefix length
IP Prefix	1-128	<p>Address prefix, in the NLRI format. The address family is used to determine whether the prefix is an IPv4 or IPv6 prefix.</p> <ul style="list-style-type: none"> In the case of an IPv4 address family, the IP Prefix field encapsulates a prefix that complies with the BGP IPv4 unicast style. In the case of an IPv6 address family, the IP Prefix field encapsulates a prefix that complies with the BGP IPv6 unicast style. Currently, only IPv6 is supported.
MIIG-Type	8	Subnet access type of an MIIG and can be single-homed access or complete multi-homed access.

Field	Length (in Bits)	Description
Flags	8	Attribute flag of the IP prefix in SPA. The current placeholder is as follows: <ul style="list-style-type: none">• S (bit 0): source flag, indicating that the IP prefix can be used as the source prefix of the origin router ID.• D (bit 1): destination flag, indicating that the IP prefix can be used as the destination prefix of the origin router ID.
MIIG-Tag	32	Subnet tag of an MIIG, which uniquely identifies an access subnet in a deployment domain.

BGP SAVNET Implementation in an Intra-domain Scenario

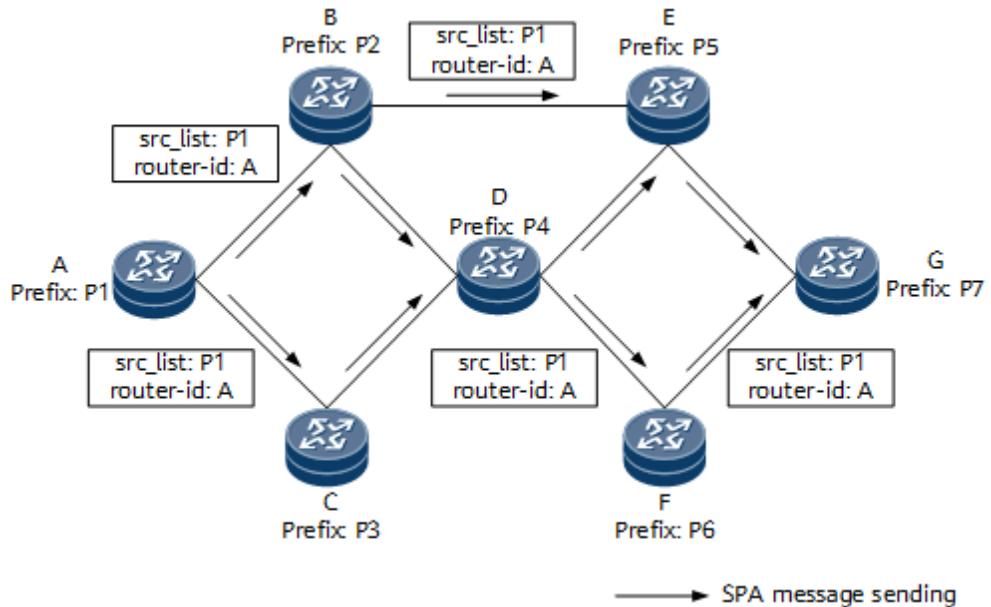
The BGP SAVNET process consists of two parts: SPA and DPP.

1. SPA

Each node advertises a list of its locally originated valid source prefixes to other nodes, as shown in [Figure 1-378](#).

- a. The origin node A obtains the list of its locally originated valid source prefixes, combines the list with its router ID to generate an SPA route, and then advertises the route to other network nodes B through G in the same AS through BGP Update messages.
- b. After receiving the SPA route, other nodes save the router ID and the list of valid source prefixes of the origin node. In addition, these nodes also function as origin nodes to advertise source prefixes. After the SPA process ends, all nodes in the AS should have the complete source prefix list of other nodes and the corresponding router ID information.
- c. When the content of the locally originated valid source prefix list changes, the SPA route needs to be updated or deleted according to the change.

Figure 1-378 Source prefix advertisement process

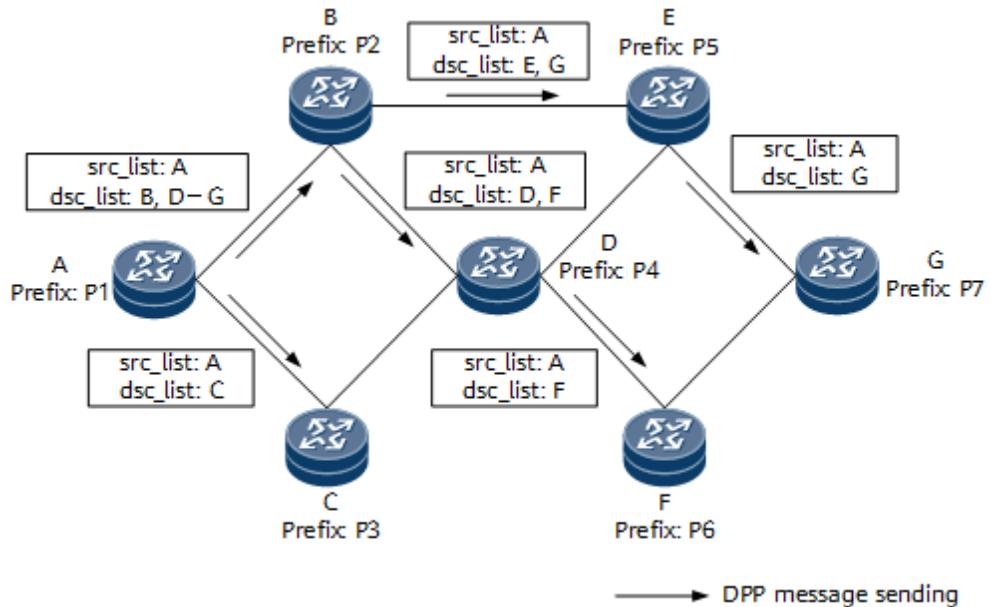


2. DPP:

DPP refers to the process in which a probing origin node sends a probing message and each relay node through which the message passes records source prefix information of the origin node and the valid inbound interface of the probing message to generate SAV rules. [Figure 1-379](#) shows this process.

- According to the foregoing rules, the origin node A initiates path probing by sending a probing message carrying its router ID and the router ID of the probing destination node G. Relay nodes (B, C, D, E, and F) relay the message according to the egress reachable destination prefix list carried in the received probing message.
- Through the SPA process, relay nodes store router IDs and valid source prefix lists of all nodes in the AS. Upon receiving a probing message, they can obtain the valid source prefix list of the origin node according to its router ID and combine the list with the inbound interface of the message to generate an SAV rule.
- A relay node determines the probing egress according to the probing destination router ID in the probing message, and forwards the message (so-called relay process). For example, relay node B determines the probing egresses B-E and B-D according to the egress reachable destination prefix list [P2, P4, P5, P6, P7]. According to next hop information, node B determines that the egress reachable destination prefix list to node E is [P5, P7] and that the egress reachable destination prefix list to node D is [P4, P6]. It can be found that the prefix P2 no longer exists in either egress reachable destination prefix list because P2 is a valid address of node B and the probing for P2 already ends on node B. Similarly, probing for the destination prefix is implemented in this hop-by-hop manner until the probing message reaches the probing destination, indicating that the probing ends.
- If the forwarding rule changes on a node, this node needs to initiate probing again.

Figure 1-379 Destination prefix probing process



NOTE

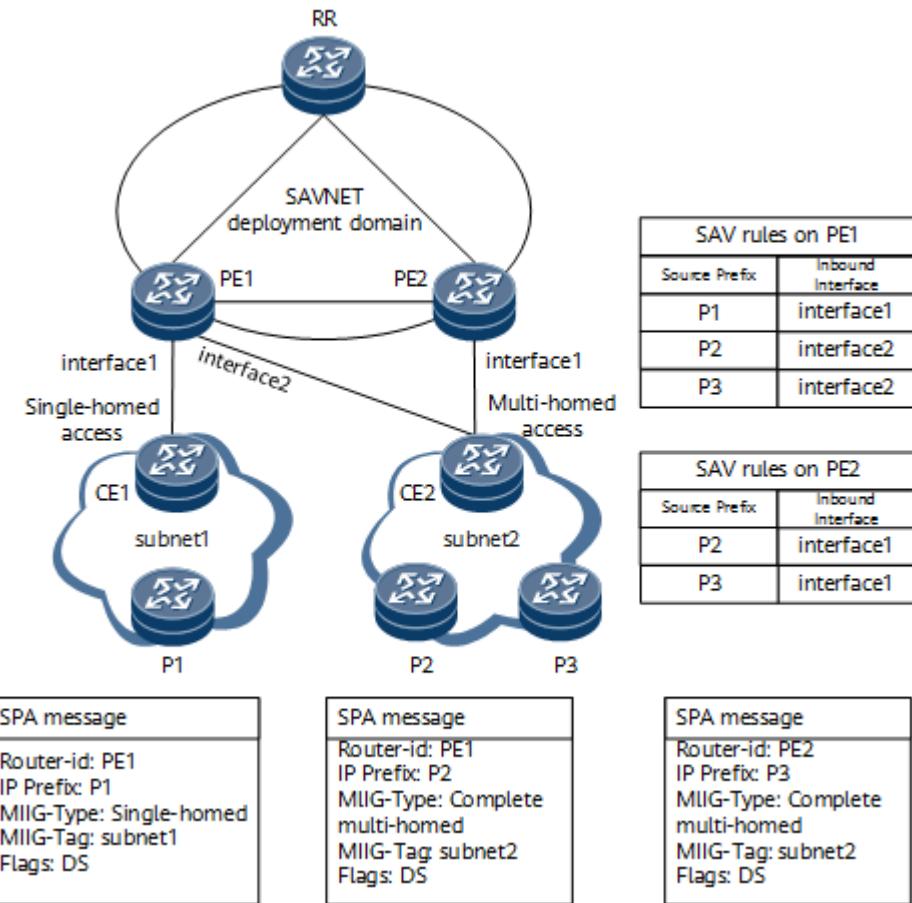
The foregoing process description is only about protocol implementation, not involving the process of performing traffic filtering according to the SAV rules generated by the protocol.

BGP SAVNET Implementation in an Access Scenario

In the access scenario, BGP SAVNET implementation requires only the SPA process to generate SAV rules.

1. The MIIG-Tag (uniquely identifying an access subnet in a deployment domain) and MIIG type (subnet access type, which can be single-homed or complete multi-homed) are configured on PE1 and PE2 interfaces connected to subnets.
2. Origin nodes PE1 and PE2 generate SPA routes for routes P1, P2, and P3 imported from subnets, add the MIIG tag and MIIG type configured on each subnet inbound interface, and add the default Source/Dest attribute based on the MIIG type.
3. After generating the SPA routes shown in [Figure 1-380](#), PE1 and PE2 advertise the SPA routes to other network nodes in the SAVNET deployment domain through BGP Update messages.
4. After a node generates or receives an SPA route, it matches the MIIG tag in the route against the local interface configured with the MIIG tag and generates an SAV rule. The forwarding plane then filters traffic based on the generated SAV rule.
5. When subnet routes change, messages for updating or withdrawing SPA routes need to be sent according to the change.

Figure 1-380 SPA route advertisement on a BGP SAVNET in an access scenario



Usage Scenario

BGP SAVNET supports valid source address validation on complex IPv6 networks. This feature applies to complex routing scenarios, such as route asymmetry, multipath, and load balancing scenarios. In access scenarios, it supports multiple subnet access types, such as single-homed and multi-homed access. This feature can effectively filter out source address forgery attack packets without compromising normal service traffic.

Benefits

Intra-domain BGP SAVNET supports the generation of an SAV table that contains the interconnection interfaces (NNIs) of intra-domain routers. SAV protection is implemented within the deployment domain, preventing the customer subnets of the deployment domain from accepting packets with forged source addresses from the local domain or other domains.

BGP SAVNET in the access scenario supports the generation of an SAV table that contains subnet inbound interfaces (UNIs) to implement the IP address whitelist function. SAV protection is implemented at the edge of the deployment domain, preventing the customer subnets of the deployment domain from accepting packets with forged source addresses from other subnets.

In terms of accuracy, deployability, and scalability, BGP SAVNET is the most suitable distributed SAV-table generation protocol in an intra-domain scenario.

1.1.9.2 BGP Configuration

Border Gateway Protocol (BGP) is applicable to complicated large-scale networks and used to transmit routing information between ASs.

1.1.9.2.1 BGP Overview

The Border Gateway Protocol (BGP) advertises and maintains a large number of routes between autonomous systems (ASs).

BGP Definition

Border Gateway Protocol (BGP) is a dynamic routing protocol used between autonomous systems (ASs).

As three earlier-released versions of BGP, BGP-1, BGP-2, and BGP-3 are used to exchange reachable inter-AS routes, establish inter-AS paths, avoid routing loops, and apply routing policies between ASs.

Currently, BGP-4 is used.

As an exterior routing protocol on the Internet, BGP has been widely used among Internet service providers (ISPs).

BGP has the following characteristics:

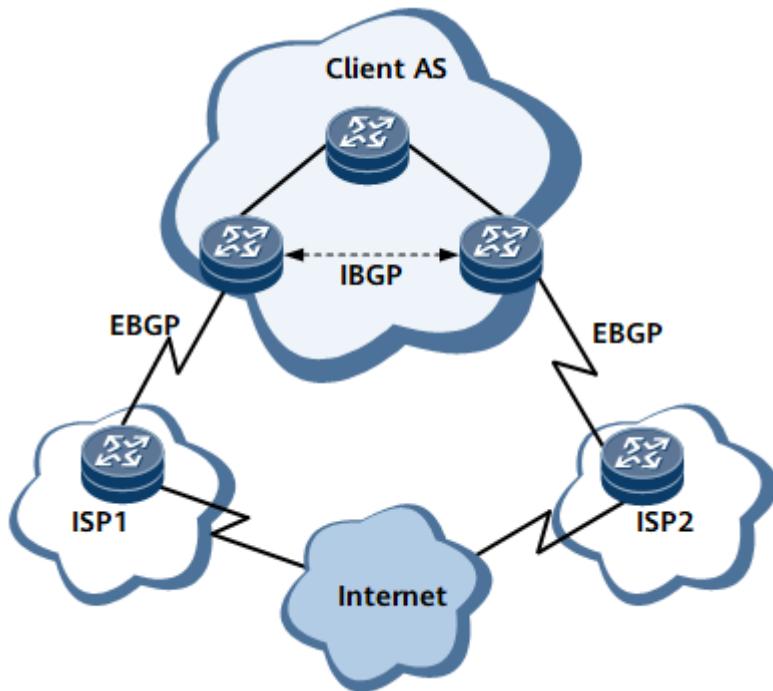
- Unlike an Interior Gateway Protocol (IGP), such as Open Shortest Path First (OSPF) and Routing Information Protocol (RIP), BGP is an Exterior Gateway Protocol (EGP) which controls route advertisement and selects optimal routes between ASs rather than discovering or calculating routes.
- BGP uses Transport Control Protocol (TCP) as the transport layer protocol, which enhances BGP reliability.
 - BGP selects inter-AS routes, which poses high requirements on stability. Therefore, using TCP enhances BGP's stability.
 - BGP peers must be logically connected through TCP. The destination port number is 179 and the local port number is a random value.
- BGP supports Classless Inter-Domain Routing (CIDR).
- When routes are updated, BGP transmits only the updated routes, which reduces bandwidth consumption during BGP route distribution. Therefore, BGP is applicable to the Internet where a large number of routes are transmitted.
- BGP is a distance-vector routing protocol.
- BGP is designed to prevent loops.
 - Between ASs: BGP routes carry information about the ASs along the path. The routes that carry the local AS number are discarded to prevent inter-AS loops.
 - Within an AS: BGP does not advertise routes learned in an AS to BGP peers in the AS to prevent intra-AS loops.
- BGP provides many routing policies to flexibly select and filter routes.
- BGP provides a mechanism for preventing route flapping, improving Internet stability.

- BGP can be easily extended to adapt to network development.

Purpose

BGP transmits route information between ASs. It, however, is not required in all scenarios.

Figure 1-381 BGP networking



BGP is required in the following scenarios:

- On the network shown in [Figure 1-381](#), users need to be connected to two or more ISPs. The ISPs need to provide all or part of the Internet routes for the users. Devices, therefore, need to select the optimal route through the AS of an ISP to the destination based on the attributes carried in BGP routes.
- The AS_Path attribute needs to be transmitted between users in different organizations.
- Users need to transmit VPN routes through a Layer 3 VPN. For details, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - VPN*.
- Users need to transmit multicast routes to construct a multicast topology. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - IP Multicast*.

BGP is not required in the following scenarios:

- Users are connected to only one ISP
- The ISP does not need to provide Internet routes for users.
- ASs are connected through default routes.

1.1.9.2.2 Configuration Precautions for BGP

Feature Requirements

Table 1-123 Feature requirements

Feature Requirements	Series	Models
Before enabling the Add-Path function, you need to set the number of paths for use according to the actual requirement. The maximum number of paths cannot be set; otherwise, memory resources may be exhausted.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
The protocol defines the prefix limit. Currently, the EVPN/VPN4/VPN6/L2VPN-AD/MVPN/SRv6 Policy IRT is saved together regardless of the address family. The implementation is the same as that in the protocol. This rule may cause all RT filters in the EVPN/VPNv4/VPNv6/L2VPN-AD/MVPN/SRv6 Policy address family to take effect.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
An EVPN RR cannot reflect routes learned from IPv4 peers to IPv6 peers. Similarly, an EVPN RR cannot reflect routes learned from IPv6 peers to IPv4 peers.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
If a feature is not configured for a peer or the default value is configured for the peer, and then the peer is added to a peer group which has a non-default value of the feature configured, the peer inherits the feature configuration from the peer group. If a peer in a peer group has the same configuration of a feature with the peer group, and then the feature is modified for the peer group, the feature configuration of the peer changes accordingly. If a peer in a peer group and the peer group have different configurations of a feature, and then the feature is modified for the peer group, the feature configuration of the peer remains inconsistent with that of the peer group.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
If the AS-SET parameter is configured during route summarization, the AS_Path attributes of all specific routes with the same sequence number form AS_SEQUENCE, and the other ASs form AS_SET, which is used as the AS_Path attribute of the summarized route. The number of AS_Path attributes after aggregation cannot exceed 2000. Otherwise, the AS_Path attribute is set to null.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
LocalIfnet does not support over GRE tunnels.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
In a scenario where a BGP IPv4 VPN unicast route is iterated to an MPLS local IFNET tunnel, the tunnel ID in the FIB table is different from that in the IP routing table. The tunnel ID in the FIB table does not carry a VPN instance ID, and the tunnel ID in the IP routing table is used.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
The peer allow-as-loop command enables BGP to check the count of the local AS number in the routes received from EBGP peers or confederation EBGP peers. The command does not apply to IBGP peers or confederation IBGP peers. If the command is not run, the implementation is equivalent to the peer allow-as-loop 0 configuration.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
EBGP peers do not support the following features: Route reflector, Best-external, Add-path. IBGP peers do not support the following features: EBGP-max-hop, MPLS local IFNET, Fake AS. Feature exclusiveness: The ebgp-max-hop and valid-ttl-hops functions are mutually exclusive.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
To advertise default routes, you need to run both the default-route imported and import-route commands. If either command is not run, default routes cannot be advertised even when they are available in the routing table.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
The undo peer x.x.x.x group command keeps the behavior of the peer unchanged before and after the operation, instead of removing the peer from the group.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
BGP routes cannot be iterated to SRv6 BE routes. After iteration, the routes are inactive.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>(1) Configuration restrictions:</p> <p>A maximum of 100 regions can be configured in a VS.</p> <p>A maximum of 100 ASs can be configured in a region, regardless of 4-byte or 2-byte ASs.</p> <p>A maximum of 100 regional confederations can be configured in a VS, and a maximum of 100 regions can be configured in a regional confederation.</p> <p>An AS cannot be added to different regions repeatedly. (The display format is not affected. For example, 1.0 and 65536 are the same AS.)</p> <p>A region cannot be added to different regional confederations repeatedly. When a region is added to a regional confederation, the region must exist.</p> <p>If a region has been added to a regional confederation, the region information in the regional confederation is also deleted when the region is deleted.</p> <p>(2) Regional validation depends on the reliability of the origin AS in the route. In actual applications, the RPKI ROA function must be deployed together with regional validation to ensure the correctness of the origin AS in the route.</p> <p>(3) The regional validation configuration is a global configuration. The scenario where the ASs of the public and private networks overlap is not considered.</p> <p>(4) Regional validation can be enabled only in the address family view. Regional validation cannot be enabled or disabled for a single peer. Currently, the following address families are supported:</p> <p>IPv4 unicast address family/IPv6 unicast address family/VPN instance IPv4 address family/VPN instance IPv6 address family/VPN instance IPv4 address family in BGP multi-instance. Labeled routes in the IPv4/IPv6 unicast address family support regional validation.</p> <p>(5) Regional validation must be configured on the border router that is directly connected to the external domain router. Internal risky routes cannot be identified.</p> <p>(6) When the import policy (for example, AS modification) is configured together with the</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X

Feature Requirements	Series	Models
<p>regional validation function, the regional validation check is performed before the import policy takes effect (after the ROA validation).</p> <p>(7) Regional validation affects the route learning performance of EBGP peers (the learning performance decreases by no more than 5% after regional validation is configured), but does not affect the convergence performance, RR reflection performance, or IBGP peer route learning performance.</p>		
<p>The route-policy with the apply cost-type med-inherit-aigp configuration takes effect on IPv4 or IPv6 VPN routes. The AIGP attributes of the VPN routes are advertised by PEs to CEs through the MED attribute. The route-policy does not take effect in other address families.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>Currently, among the 11 types of segments specified in the protocol, only Type 1 SIDs (MPLS label SIDs) and Type 2 SIDs (IPv6 address SIDs) are supported.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>For a peer in the SR-MPLS TE Policy address family, if the configured routing policy contains if-match ip-prefix/acl/rd-filter, the route is matched. This is because the IP prefix, ACL, or RD filter cannot be used for matching for NLRI in the SR-MPLS TE Policy address family. If the if-match as-path or cost command is configured in the routing policy, the routing policy is matched only when the matching conditions are met.</p> <p>You are advised to properly plan routing policies and do not use unsupported matching modes.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>Only BGP IPv4 public network unicast routes, BGP IPv4 VPN remotely leaked routes, BGP IPv4 public network labeled routes (6PE), and BGP IPv6 VPN remotely leaked routes can be recursed to SR-MPLS TE Policies using nexthop +color.</p> <p>You are advised to plan tunnel-policy and tunnel-selector properly.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
By default, static routes cannot be iterated to SRv6 BE routes. After the ip route-static recursive-lookup inherit-label-route segment-routing-ipv6 command is run, static routes can be iterated to SRv6 BE routes.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
BGP IPv4 public network unicast routes, BGP IPv4 VPN remote cross routes, BGP IPv6 public network labeled routes, and BGP IPv6 VPN remote cross routes that can be recursed to SR-MPLS TE Policy tunnels carry multiple color extended community attributes, only the largest color value is used to recurse routes to SR-MPLS TE Policy tunnels. If the CO flag (defined in draft-ietf-idr-segment-routing-te-policy) of a received route is any value, the route is processed as 00. You are advised to properly plan the color attribute added to routes on the egress node.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>Hackers usually use the source addresses of other users to launch DDoS attacks. URPF check on the source addresses of packets at the network edge can effectively prevent such attacks and save network bandwidth. After receiving a packet, the router checks the source address of the packet and the outbound interface of the packet. If the outbound interface of the packet is the same as the inbound interface of the packet, the router considers the packet valid. Otherwise, the router considers the packet invalid and discards it.</p> <p>URPF based on BGP route source interfaces has the following restrictions:</p> <ol style="list-style-type: none">1. Only IPv4 and IPv6 unicast public networks are supported.2. The source route received by the peer from interface A is different from interface B that guides traffic route recursion. As a result, interface A that receives the source route is different from interface B that receives traffic (for example, the next hop is changed based on a policy, an indirectly connected BGP peer, or an RR). Users need to analyze the scenario to add or delete configurations.3. Only EBGP scenarios are supported.4. The same peer ID must be set for all outbound interfaces to which the peer address can be recursed. Otherwise, the interface that receives the route is different from the interface that receives the traffic. As a result, the traffic is discarded.5. Only one peer ID can be configured on the current interface. Therefore, if different peers use the same interface, the same policy must be enabled on the peers (the peer IDs in the policy are the same).6. After the peer ID is configured (the corresponding policy contains QoS information), the ipv6 qppb enable command is not configured. However, when QPPB is enabled globally, QoS takes effect.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
The function that DX4 SIDs are applied for BGP public network IPv4 routes based on the next hop and the add-path function are mutually exclusive. If the add-path function is enabled first, a DX4 SID will not be applied for. If a DX4 SID is obtained before the add-path function is enabled, the obtained DX4 SID is released.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
After a route is imported between VPN and public network instances, the next hop or color extended community attribute of the route cannot be changed through a route-policy. Properly plan the route-policy to be used during the route import between VPN and public network instances.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
IPv4 and IPv6 neighbor functions cannot be enabled at the same time in the VPNv4 address family.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X

Feature Requirements	Series	Models
<p>Scenario 1: The as-number command is run in the BGP-VPN instance IPv4 address family view and BGP-VPN instance IPv6 address family view to configure a private AS number, and the bgp yang-mode enable command is run to configure the YANG management mode for the BGP VPN instance.</p> <p>Restrictions:</p> <ol style="list-style-type: none">1. After the bgp yang-mode enable command is run, the sequence of the peer as-number and as-number commands in the configuration file changes. As a result, the configurations fail to be pasted. To solve this problem, configure a private AS number and then run other commands.2. BGP peer relationships cannot be enabled in the BGP-Flow VPN instance IPv4 address family view, BGP-Flow VPN instance IPv6 address family view, or BGP-Labeled-VPN instance IPv4 address family view. VPN IPv4 peers cannot be enabled in the BGP-VPN instance IPv6 address family view.3. Before deleting a VPN AS number, delete the peers or peer groups in the VPN address family and the peers in the BGP-VPN instance view.4. Before running the undo bgp yang-mode enable command, delete the private AS number. <p>Scenario 2: A private AS number is configured in the BGP-VPN instance IPv4 address family view and BGP-VPN instance IPv6 address family view, and the YANG management mode is configured for the BGP VPN instance, and then the YANG management mode is disabled.</p> <p>Restrictions:</p> <ol style="list-style-type: none">1. After the bgp yang-mode enable command is run, the sequence of the peer as-number and as-number commands in the configuration file changes. As a result, the configurations fail to be pasted. You can run the peer as-number command in the BGP-VPN instance IPv4 address family view or BGP-VPN instance IPv6 address family view to configure a new peer or peer group. However, you cannot configure a new peer or peer group in the BGP-VPN instance view.2. BGP peer relationships cannot be enabled in the BGP-Flow VPN instance IPv4 address family	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>view, BGP-Flow VPN instance IPv6 address family view, or BGP-Labeled-VPN instance IPv4 address family view. VPN IPv4 peers cannot be enabled in the BGP-VPN instance IPv6 address family view.</p> <p>3. Before deleting a private AS number, delete the peers or peer groups in the VPN address family and the peers in the BGP-VPN instance view.</p> <p>Note: The address families listed in this restriction are all restricted BGP address families. The actual address families depend on the address family capabilities supported by the device.</p>		
<p>Pay attention to the following points when using BGP EPE:</p> <ol style="list-style-type: none"> 1. Confederation Member ASN is not supported. 2. Peer groups cannot be enabled. 3. Only node labels are generated for directly connected EBGP peers. Adjacency labels are not generated. 4. BGP EPE takes effect only after the BGP-LS address family is enabled. 5. BGP EPE supports only two directly connected devices, and multi-hop EBGP supports only two directly connected devices. 6. BGP EPE label forwarding takes effect only after segment routing is enabled in the system view. 7. When the next hop to which the BGP EPE peer address is recursed is an IPv6 address, no link route is generated. 	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>After the peer fake-as command is run and the prepend-fake-as or prepend-global-as parameter is modified, the BGP peer relationship is reestablished.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
If IPv6 routes received from IPv6 peers recurse to 6PE routes, the Pop-Go forwarding mode is not supported.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
The second carrier accesses the first carrier in IGP+LDP mode. Load balancing or FRR from the second carrier to the first carrier is not supported: 1->n+1.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
The original community value contained in all BGP public network routes on the device is different from the community value configured for the peer. Otherwise, the original community will be replaced or routes will be incorrectly filtered. The community value must be unique and depends on the overall network planning.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
During BGP peer relationship establishment, specifying the local address is only optional. Therefore, when binding a TWAMP Light test instance to a connection based on the virtual link of the peer, the device cannot check whether the local and remote addresses of the peer match those of the TWAMP Light test instance. However, the generated link routes do not contain delay information.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
In BGP EPEv6 scenarios, only End/End.X SIDs support compression.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X
UNRs do not support POPGO.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X

Feature Requirements	Series	Models
<p>1. If a device on the looped path does not support loop detection or is not enabled with loop detection, the low priority of the looped route may be lost. As a result, the loop cannot be eliminated.</p> <p>Workaround: Ensure that loop detection is enabled on each device on the looped path and the number of devices on the looped path does not exceed four.</p> <p>Impact: The loop detection function does not take effect.</p> <p>2. If the local preference or MED of a looped route is reduced and then changed to another value using a route-policy, the priority of the looped route cannot be reduced. As a result, the loop cannot be eliminated.</p> <p>Workaround: Do not use a route-policy to modify the local-preference or MED of the looped route.</p> <p>Impact: The loop detection function does not take effect.</p> <p>3. If the priority of the original route is the lowest, the priority cannot be reduced to eliminate the loops.</p> <p>Workaround: None</p> <p>Impact: The loop detection function does not take effect.</p> <p>4. After the clear route loop-detect bgp alarm command is executed, the switchover is performed immediately. Some looped route records may not be cleared. In this case, you need to run the command again to clear the alarm.</p> <p>Workaround: None</p> <p>Impact: Even if a route does not carry the loop attribute, the route is incorrectly considered as a looped route. As a result, the route is never selected.</p> <p>5. The maximum number of looped routes that can be recorded is controlled by the PAF file. Excess looped routes are not recorded. As a result, loops may fail to be removed for some routes.</p> <p>Workaround: None</p> <p>Impact: The loop detection function does not take effect.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
<p>6. The loop attribute values of BGP/IS-IS/OSPF are randomly generated. There is a low probability that the random numbers of different devices are the same. As a result, the loop attributes of BGP and BGP, IS-IS/OSPF and IS-IS/OSPF, and BGP and IS-IS/OSPF are the same, which may cause misjudgment of looped routes.</p> <p>Workaround: None</p> <p>Impact: Looped routes are incorrectly determined.</p> <p>7. If both a source route and a looped route exist on a device and the source route is deleted, the looped route may be preferentially selected. In this case, enabling loop detection cannot prevent loops, and looped routes cannot be withdrawn.</p> <p>Workaround: If both source routes and looped routes exist on a device, remove the loop first and then delete the routes.</p> <p>Impact: The looped route flaps on the looped path, and the route cannot be withdrawn after a loop occurs.</p>		
<p>If no public SID is carried, SRv6 BE escape cannot be performed when BGP public network IPv6 over SRv6 TE Policy/SRv6 TE Flow Group (USD-capable tunnels) are used.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X
<p>ROA verification is not performed on the routes received from IBGP peers and locally imported routes.</p>	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

1.1.9.2.3 Configuring Basic BGP Functions

Before building a BGP network, you must configure basic BGP functions.

Usage Scenario

BGP can be configured on a network to implement communication among ASs. To build a BGP network, configure basic BGP functions, including the following steps:

- Start a BGP process. This step is a prerequisite for configuring basic BGP functions.
- Establish BGP peer relationships. Devices can exchange BGP routing information only after peer relationships are established.
- Import routes: BGP itself cannot discover routes. Instead, it needs to import routes discovered by other protocols to generate BGP routes.

NOTE

Commands in the BGP-IPv4 unicast address family view can be run in the BGP view, which facilitates configuration. These commands are displayed in the BGP-IPv4 unicast address family view in configuration files.

To prevent commands of the BGP-IPv4 unicast address family from being executed in the BGP view, run the **bgp default ipv4-unicast-config disable** command.

Pre-configuration Tasks

Before configuring basic BGP functions, configure parameters of the link layer protocol and IP addresses for interfaces to ensure that the link layer protocol on the interfaces is Up.

Starting a BGP Process

Starting a BGP process is a prerequisite for configuring BGP functions. When starting a BGP process on a device, you need to specify the number of the AS to which the device belongs.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 (Optional) Run **route loop-detect bgp enable**

BGP routing loop detection is enabled.

After this function is enabled, the device reports an alarm when detecting a BGP routing loop. Because the device cannot determine whether a routing loop is removed, the alarm will not be cleared automatically even if the routing loop is removed. To manually clear the BGP routing loop alarm, run the **clear route loop-detect bgp alarm** command.

Step 3 Run **bgp as-number**

BGP is started (with the local AS number specified), and the BGP view is displayed.

Step 4 (Optional) Run **router-id ipv4-address**

A BGP router ID is configured.

Configuring a new BGP router ID or changing the existing one will reset the BGP peer relationship between routers.

If two devices have different router IDs, an IBGP or EBGP connection can be established between them. If the router IDs are the same and the **router-id allow-same enable** command is run, an EBGP connection can be established.

 NOTE

By default, BGP automatically selects a router ID configured in the system view as its router ID. If the selected router ID is the IP address of a physical interface, the change in the IP address will cause route flapping. To enhance network stability, configure the IP address of a loopback interface as the router ID. For rules in selecting router IDs from the system view, see the **router-id** command description in the "Command Reference."

By default, Cluster_List takes precedence over router ID during BGP route selection. To enable router ID to take precedence over Cluster_List during BGP route selection, run the **bestroute routerid-prior-clusterlist** command.

Step 5 (Optional) Run **shutdown**

All sessions between the device and its BGP peers are terminated.

Frequent BGP route flapping may occur during system upgrade and maintenance. To prevent this from impacting the network, perform this step.

 NOTICE

After an upgrade or maintenance, run the **undo shutdown** command to restore the BGP sessions; otherwise, BGP functions will not work properly.

Step 6 Run **commit**

The configuration is committed.

----End

Configuring a BGP Peer

Devices can exchange BGP routing information only after the BGP peer relationship is established.

Context

BGP peers use TCP to establish connections. Therefore, you need to specify the IP addresses of the peers when configuring BGP. A BGP peer may not be a neighboring node, and the BGP peer relationship can be created through logical links. To enhance the stability of BGP connections, using loopback interface addresses to establish connections is recommended.

The devices in the same AS establish IBGP peer relationships, and the devices of different ASs establish EBGP peer relationships.

Procedure

- Configure an IBGP peer.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **peer { ipv4-address | peerGroupName } as-number as-number**

The IP address of the peer and the number of the AS where the peer resides are specified.

The number of the AS where the specified peer resides must be the same as that of the local AS.

The IP address of the peer can be one of the following types:

- IP address of the peer's interface that is directly connected to the local device
 - IP address of a sub-interface on a directly connected peer
 - IP address of a reachable loopback interface on the peer
- d. (Optional) Run **peer { ipv4-address | peerGroupName } connect-interface** *interface-type interface-number [ipv4-source-address]*

The source interface and source address are specified for establishing a TCP connection with the specified BGP peer.

NOTE

If the IP address of a loopback interface or a sub-interface is used to establish a BGP connection, you are advised to run the **peer connect-interface** command at both ends of the connection to ensure that the connection is correctly established. If this command is run on only one end, the BGP connection may fail to be established.

If the source interface used by the local end to establish a BGP session is configured with multiple IP addresses, the **peer connect-interface** command must be run to specify the source address. This ensures the correct connection between the two ends.

- e. (Optional) Run **peer { ipv4-address | peerGroupName } description** *description-text*

A description is configured for the peer.

To simplify network management, you can configure a description for a specified peer.

- f. (Optional) Run **peer { ipv4-address | peerGroupName } tcp-mss** *tcp-mss-number*

A TCP MSS value used during TCP connection establishment with a peer or peer group is configured.

This configuration ensures that TCP packets can still be segmented properly based on the specified TCP MSS in cases where the path MTU is unavailable, thereby improving network performance.

- g. Run **commit**

The configuration is committed.

- Configure an EBGP peer.

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

c. (Optional) Run **router-id allow-same enable**

The peers with the same router ID are allowed to establish EBGP connections.

d. Run **peer { ipv4-address | peerGroupName } as-number as-number**

A peer IP address and the number of the AS where the peer resides are specified.

The number of the AS where the specified peer resides must be different from that of the local AS.

The IP address of the peer can be one of the following types:

- IP address of the peer's interface that is directly connected to the local device
- IP address of a sub-interface on a directly connected peer
- IP address of a reachable loopback interface on the peer

e. (Optional) Run **peer { ipv4-address | peerGroupName } connect-interface interface-type interface-number [ipv4-source-address]**

The source interface and source address are specified for establishing a TCP connection with the specified BGP peer.

 NOTE

If the IP address of a loopback interface or a sub-interface is used to establish a BGP connection, you are advised to run the **peer connect-interface** command at both ends of the connection to ensure that the connection is correctly established. If this command is run on only one end, the BGP connection may not be established.

f. (Optional) Run **peer { ipv4-address | peerGroupName } ebgp-max-hop [hop-count]**

The maximum number of hops allowed over an EBGP connection is set.

In most cases, EBGP peers are directly connected through a physical link. If you want to establish an EBGP peer relationship between indirectly connected peers, run the **peer ebgp-max-hop** command to set the maximum number of hops allowed over the TCP connection.

 NOTE

If loopback interfaces are used to establish an EBGP connection, run the **peer ebgp-max-hop** command, where the value of *hop-count* must be greater than or equal to 2. Otherwise, the EBGP peer relationship cannot be established.

g. (Optional) Run **peer { ipv4-address | peerGroupName } description description-text**

A description is configured for the peer.

To simplify network management, you can configure a description for a specified peer.

h. (Optional) Run **peer { ipv4-address | peerGroupName } peer-as-check**

The device is configured not to advertise the routes learned from an EBGP peer to the peers in the same AS with the EBGP peer.

By default, after receiving a route from an EBGP peer (for example, in AS 200), the local device (for example, in AS 100) advertises the route to other EBGP peers in AS 200. After the **peer peer-as-check** command is run on the device, the device does not advertise the routes received from an EBGP peer to other EBGP peers in the same AS with this EBGP peer. This reduces BGP memory and CPU consumption and speeds up route convergence in case of route flapping.

- i. (Optional) Run **peer { ipv4-address | peerGroupName } tcp-mss *tcp-mss-number***

A TCP MSS value used during TCP connection establishment with a peer or peer group is configured.

This configuration ensures that TCP packets can still be segmented properly based on the specified TCP MSS in cases where the path MTU is unavailable, thereby improving network performance.

- j. Run **commit**

The configuration is committed.

----End

Configuring BGP to Import Routes

BGP can import the routes from other routing protocols. When routes are imported from dynamic routing protocols, the process IDs of the routing protocols must be specified. Importing routes from other protocols can enrich the BGP routing table. When importing IGP routes, BGP can filter the routes by routing protocol.

Context

BGP itself cannot discover routes. It needs to import routes from other protocols (such as IGPs) to the BGP routing table so that the routes can be transmitted within an AS or between ASs. When importing routes, BGP can filter these routes by routing protocol.

BGP imports routes in the following modes:

- Import mode: BGP imports routes into its routing table by protocol type, such as RIP, OSPF, IS-IS, static routes, or direct routes.
- Network mode: BGP imports a route with the specified prefix and mask. This mode is more precise than the Import mode.

Procedure

- Import mode
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp *as-number***
The BGP view is displayed.

c. (Optional) Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

d. Run **import-route { isis process-id | ospf process-id | rip process-id | direct | static | unr } [[med med] | [route-policy route-policy-name] | [route-filter route-filter-name]] * [non-relay-tunnel]**

BGP is configured to import routes from another protocol.

To set an MED value for the imported routes, specify the *med* parameter. An EBGP peer selects the route with the lowest MED value to guide traffic entering the AS where the peer resides.

To filter the routes imported from another protocol, you can specify the **route-policy route-policy-name** parameter.

To filter the routes imported from another protocol, you can also specify the **route-filter route-filter-name** parameter.

If **non-relay-tunnel** is specified, the routes imported by BGP do not recurse to tunnels. Generally, the routes imported by BGP can be recursed to tunnels. However, in some scenarios, if the routes imported by BGP are recursed to tunnels, problems may occur. For example, in a seamless MPLS scenario, egress protection is configured between egress MASGs, and a tunnel exists between the MASGs. If a BGP route imported by an MASG is recursed to the tunnel, the label action is not popping out but stitching tunnels. Consequently, traffic is diverted to the peer MASG, which prolongs the traffic switching time. As a result, egress protection fails to take effect. To solve this problem, specify the **non-relay-tunnel** parameter to prevent the routes imported by BGP from being recursed to tunnels. This prevents tunnel stitching, which would otherwise cause egress protection to become invalid.

 **NOTE**

When configuring the device to import the routes discovered by a dynamic routing protocol, such as OSPF, RIP, or IS-IS, specify the process ID of the protocol.

In a BAS device access scenario, to prevent BGP from importing the UNRs that cannot be advertised, run the **access no-advertise-unr import disable** command.

e. (Optional) Run **default-route imported**

BGP is allowed to import default routes from the local IP routing table. BGP can import default routes only when both the **default-route imported** and **import-route** commands have been run. Using only the **import-route** command is insufficient to import default routes, and using only the **default-route imported** command imports only the default routes that exist in the local routing table.

f. Run **commit**

The configuration is committed.

- Network mode

- a. Run **system-view**

- The system view is displayed.
- b. Run **bgp as-number**
- The BGP view is displayed.
- c. (Optional) Run **ipv4-family unicast**
- The BGP-IPv4 unicast address family view is displayed.
- d. Run **network ipv4-address [mask | mask-length] [route-policy route-policy-name | route-filter route-filter-name] [non-relay-tunnel]**
- BGP is configured to import local routes.
- If no mask or mask length is specified, the address is processed as a classful address. The local routes to be imported must exist in the local IP routing table.
- To use a route-policy to filter the routes to be imported, specify the **route-policy route-policy-name** parameter.
- To use a route-filter to filter the routes to be imported, specify the **route-filter route-filter-name** parameter.
- If **non-relay-tunnel** is specified, the routes imported by BGP do not recurse to tunnels. Generally, the routes imported by BGP can be recursed to tunnels. However, in some scenarios, if the routes imported by BGP are recursed to tunnels, problems may occur. For example, in a seamless MPLS scenario, egress protection is configured between egress MASGs, and a tunnel exists between the MASGs. If a BGP route imported by an MASG is recursed to the tunnel, the label action is not popping out but stitching tunnels. Consequently, traffic is diverted to the peer MASG, which prolongs the traffic switching time. As a result, egress protection fails to take effect. To solve this problem, specify the **non-relay-tunnel** parameter to prevent the routes imported by BGP from being recursed to tunnels. This prevents tunnel stitching, which would otherwise cause egress protection to become invalid.

NOTE

- The specified route can be imported only when the destination address and mask specified in the **network** command are the same as those in the corresponding entry in the local IP routing table.
- When using the **undo network** command to clear the existing configuration, specify the correct mask.

- e. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring the basic BGP functions, verify BGP peer information.

Prerequisites

Basic BGP functions have been configured.

Procedure

- Run the **display bgp router-id [vpn-instance [vpn-instance-name]]** command to check the router ID of the router.
- Run the **display bgp peer [verbose]** command to check the information about all BGP peers.
- Run the **display bgp peer ipv4-address { log-info | verbose }** command to check the information about a specified BGP peer.
- Run the **display bgp routing-table** command to check the information about BGP routes.
- Run the **display bgp routing-table route-filter route-filter-name** command to check BGP routes that match a specified route-filter.

----End

1.1.9.2.4 Configuring the Format of BGP 4-Byte AS Numbers

You can change the format of BGP 4-byte AS numbers to your desired format.

Usage Scenario

By default, a BGP 4-byte AS number is displayed in dotted notation. If you prefer 4-byte AS numbers in integer format, perform this configuration task to change the display format of 4-byte AS numbers from dotted notation to integer.

NOTICE

Changing the format of 4-byte AS numbers will affect matching results of AS_Path regular expressions and extended community attribute filters. Therefore, if the system is using an AS_Path regular expression or an extended community attribute filter as an import or export policy, you need to reconfigure the AS_Path regular expression using the **ip as-path-filter** command or the extended community attribute filter using the **ip extcommunity-filter** or **ip extcommunity-list soo** command after changing the format of 4-byte BGP AS numbers. Otherwise, routes cannot match the export or import policy, and a network fault occurs.

- If 4-byte AS numbers in integer format are configured, you need to change the format of 4-byte AS numbers in AS_Path regular expressions or extended community filters to integer accordingly.
- If 4-byte AS numbers in dotted notation are configured, you must change 4-byte AS numbers in AS_Path regular expressions and extended community attribute filters to 4-byte AS numbers in dotted notation.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **as-notation plain**

The display format of BGP 4-byte AS numbers is set to the integer format.

 NOTE

The **as-notation plain** command affects the display format of BGP 4-byte AS numbers in the outputs of display commands and the matching results of AS_Path regular expressions and extended community filters, but does not affect the configuration format.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

After configuring the format of BGP 4-byte AS numbers, perform the following operations to check the configuration:

- Run the **display bgp peer [[ipv4-address] verbose]** command to check the format of BGP 4-byte AS numbers.
- Run the **display bgp routing-table** command to check whether the result of matching BGP routes against an AS_Path regular expression or extended community filter has changed. If a change has occurred, reconfigure the AS_Path regular expression or extended community filter in time.

1.1.9.2.5 Configuring BGP Route Attributes

Configuring BGP route attributes can change BGP route selection results.

Usage Scenario

BGP has many route attributes. You can change route selection results by configuring attributes for routes.

- **BGP priority**
Setting the BGP priority can control route selection between BGP routes and routes of other routing protocols.
- **Preferred values**
After preferred values are set for BGP routes, the route with the greatest value is preferred when multiple routes to the same destination exist in the BGP routing table.
- **Local_Pref**
The Local_Pref attribute has the same function as the preferred value of a route. If both of them are configured for a BGP route, the preferred value takes precedence over the Local_Pref attribute.
- **MED**
The MED attribute is used to determine the optimal route for traffic that enters an AS. The route with the smallest MED value is selected as the optimal route if the other attributes of the routes are the same.
- **Next_Hop**
BGP route selection can be controlled by changing Next_Hop attributes for routes.
- **AS_Path**

The AS_Path attribute is used to prevent routing loops and control route selection.

- AIGP

The AIGP attribute ensures that all devices in an AIGP domain forward data along the optimal path.

Pre-configuration Tasks

Before configuring BGP route attributes, [configure basic BGP functions](#).

Setting the BGP Preference

Setting the BGP preference can affect route selection between BGP routes and other routing protocols' routes.

Context

Multiple dynamic routing protocols can run on a device. The system sets a default preference for each routing protocol for inter-protocol route sharing and selecting. If different protocols have routes to the same destination, the route with the highest preference is selected.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **preference { external internal local | route-policy route-policy-name | route-filter route-filter-name }**

A preference is set for BGP.

The lower the value, the higher the preference.

BGP has the following types of routes:

- EBGP routes: routes learned from external peers
- IBGP routes: routes learned from internal peers
- Locally originated BGP routes: summary routes generated automatically using the **summary automatic** command or manually using the **aggregate** command

You can set different preferences for the three types of routes.

You can also set a preference only for the routes that match the filtering rules in a route-policy. The routes that do not match the filtering rules will use the default preference.

 NOTE

Currently, the **peer route-policy** or **peer route-filter** command cannot be used to set a preference for the BGP routes received from or advertised to a specified peer.

Step 5 Run **commit**

The configuration is committed.

----End

Setting a PrefVal for BGP Routes

After PrefVal values are set for routes, the route with the largest PrefVal is preferred when multiple routes to the same destination exist in the BGP routing table.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **peer { group-name | ipv4-address } preferred-value preferredvalue**

A PrefVal is set for all the routes learned from a specified peer.

After the **peer preferred-value** command is run, all the routes learned from a peer have the same PrefVal.

Step 5 Run **commit**

The configuration is committed.

----End

Setting the Default Local_Pref Attribute for the Local Device

The function of the Local_Pref attribute is similar to that of the preferred value. The priority of the Local_Pref attribute, however, is lower than that of the preferred value.

Context

The Local_Pref attribute is used to determine the optimal route for the traffic that leaves an AS. When a BGP device obtains multiple routes to the same destination address but with different next hops from different IBGP peers, the BGP device prefers route with the largest Local_Pref.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **default local-preference local-preference**

The default Local_Pref attribute is set for the local device.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring MED Attributes for BGP Routes

The MED attribute is equivalent to the metric used by an IGP. The MED attribute determines the optimal route for the traffic entering an AS.

Context

If the router running BGP obtains multiple routes from different EBGP peers and these routes have the same destination but different next hops, the router selects the route with the smallest MED value when other conditions are the same.

Procedure

- Set the default MED value on a device.

- Run **system-view**

The system view is displayed.

- Run **bgp as-number**

The BGP view is displayed.

- Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

- Run **default med med**

A default MED value is configured.



The **default med** command is valid only for routes imported using the **import-route** command and summary BGP routes on the local router.

- Run **commit**

The configuration is committed.

- Compare the MED values of the routes from different ASs.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bgp as-number**

The BGP view is displayed.
 - c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.
 - d. Run **compare-different-as-med**

The MED values of routes from different ASs are compared.

By default, the BGP device compares the MED values of only routes from different peers in the same AS.
 - e. Run **commit**

The configuration is committed.
- Configure the deterministic-MED function.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bgp as-number**

The BGP view is displayed.
 - c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.
 - d. Run **deterministic-med**

The deterministic-MED function is enabled. If the deterministic-MED function is not enabled and the device receives multiple routes with the same prefix from different ASs, the sequence in which routes are received determines the route selection. After the deterministic-MED function is enabled, these routes are first grouped based on the leftmost AS number in the AS_Path attribute. Routes with the same leftmost AS number are grouped together and compared, and an optimal route is selected in the group. The optimal route in this group is then compared with the optimal routes from other groups to determine the final optimal route. With the deterministic-MED function, the route selection result is independent of the sequence in which routes are received.
 - e. Run **commit**

The configuration is committed.
- Configure the processing mode when the MED value is lost.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bgp as-number**

The BGP view is displayed.
 - c. Run **ipv4-family unicast**

- The IPv4 unicast address family view is displayed.
- d. Run **bestroute med-none-as-maximum**

The maximum MED value is used as the MED when a route carries no MED.

If the **bestroute med-none-as-maximum** command is not run and a route carries no MED, 0 is used as the MED value of the route.
 - e. Run **commit**

The configuration is committed.
- Compare the MED values of routes in a confederation.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bgp as-number**

The BGP view is displayed.
 - c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.
 - d. Run **bestroute med-confederation**

The MED values of routes in a confederation are compared.
 - e. Run **commit**

The configuration is committed.
 - Compare the sums of MED multiplied by a MED multiplier and IGP cost multiplied by an IGP cost multiplier.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bgp as-number**

The BGP view is displayed.
 - c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.
 - d. Run **bestroute med-plus-igp [igrp-multiplier igrp-multiplier | med-multiplier med-multiplier]**

The sums of MED multiplied by a MED multiplier and IGP cost multiplied by an IGP cost multiplier are compared.
 - e. Run **commit**

The configuration is committed.
 - Enable BGP to remove the MED attribute from the imported routes that are locally leaked and are to be advertised to a specified peer after an export route-policy in which the **apply cost-type** command is run is applied to routes to be advertised to the peer.
 - a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **local-cross-routing non-med**

BGP is configured to remove the MED attribute from the imported routes to be advertised to a peer after the routes are locally leaked in the scenario where an export policy with the **apply cost-type** command configured is applied to the peer.

d. Run **commit**

The configuration is committed.

----End

Configuring the Next_Hop Attribute

Configuring the Next_Hop attribute allows for flexible control of BGP route selection.

Procedure

- Configure the device to change the next hop address of a route when the device advertises the route to an IBGP peer.

When an ASBR learns a route from an EBGP peer and forwards the route to IBGP peers, the ASBR does not change the next hop of the route by default. However, the next hop of the route sent by the EBGP peer is the peer address of the EBGP peer. After the IBGP peers in the AS to which the local peer belongs receive the route, the route is inactive because its next hop is unreachable. To address this problem, configure the ASBR to change the next hop of the route learned from an EBGP peer to the ASBR's own address before the ASBR advertises the route to an IBGP peer. As an IGP runs within the AS, the next hop of the route is reachable to the IBGP peer. As such, the route received by the IBGP peer is active.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } next-hop-local**

The device is configured to change the next hop address of a route to the local address before the device advertises the route.

 NOTE

If BGP load balancing is configured, the local device changes the next hop address of a route to the local address when advertising the route to an IBGP peer or peer group, regardless of whether the **peer next-hop-local** command is run.

e. Run **commit**

The configuration is committed.

- Prevent a device from changing the next hop address of a route imported from an IGP when the device advertises the route to an IBGP peer.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

- To configure the device not to change the next hop address of an imported IGP route when the device advertises the IGP route, run either of the following commands as required:

Run the **peer { ipv4-address | group-name } next-hop-invariable** command to configure the device not to change the next hop address of an imported IGP route when the device advertises the route to the specified peer.

Run the **peer { ipv4-address | group-name } next-hop-invariable include-static-route** command to configure the BGP speaker to retain the original next hop address of an imported static route when it advertises the static route to the specified IBGP peer. However, the local interface address is used as the next hop in the following situations: (1) The imported static route is a public network static route, and its original next hop is invalid. (2) The imported static route is a public network static route, and its original next hop recurses to a VPN route. (3) The imported static route is a VPN static route and is imported from the public network instance.

Run the **peer { ipv4-address | group-name } next-hop-invariable include-unicast-route** command to configure the BGP speaker not to change the next hop address when advertising a unicast route learned from a peer to the specified EBGP peer.

e. Run **commit**

The configuration is committed.

- Prevent an ASBR from changing the next hop address of a route when the ASBR advertises the route to an EBGP peer.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family vpng4 [unicast]**

The BGP-VPNg4 address family view is displayed.

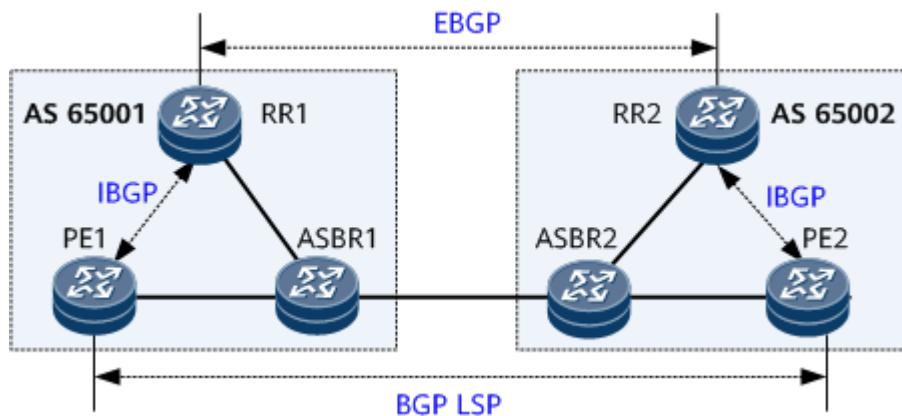
d. Run **peer { group-name | ipv4-address } next-hop-invariable**

The device is configured not to change the next hop address of a route before the device advertises the route to an EBGP peer.

In an inter-AS VPN Option C scenario where an RR is deployed, the **peer next-hop-invariable** command needs to be run on the RR to prevent the RR from changing the next hop address of a route before the RR advertises the route to an EBGP peer. This ensures that the remote PE forwards traffic through the BGP LSP destined for the local PE.

On the network shown in [Figure 1-382](#), a BGP LSP is established between PE1 and PE2. VPNv4 routes are exchanged through BGP-VPNv4 peer relationships established between PE1 and RR1, between RR1 and RR2, and between RR2 and PE2.

Figure 1-382 Inter-AS VPN Option C networking with RRs deployed



If PE1 needs to advertise a VPNv4 route to PE2, the following process is implemented:

- i. PE1 advertises the route to RR1, and the route next hop is PE1.
- ii. Upon receipt, RR1 changes the route next hop to itself and then advertises the route to RR2 through the EBGP peer relationship.
- iii. RR2 advertises the received route to its IBGP peer PE2. By default, the router changes the next hop address of a labeled route received from an EBGP peer before advertising the labeled route to an IBGP peer. Therefore, RR2 changes the next hop address of the route to its own address before advertising the route to PE2.

The next hop of the VPNv4 route received by PE2 is RR2. However, the destination of the BGP LSP is PE1. As a result, the VPNv4 route on PE2 cannot recurse to the BGP LSP, causing traffic interruption.

To solve the preceding problem, run the **peer next-hop-invariable** command on RR1 to ensure that RR1 does not change the next hop address when advertising routes to RR2. In addition, run the **peer next-hop-invariable** command on RR2 to ensure that the next hop of the route advertised by RR2 to PE2 is not changed. In this way, the next hop of the route received by PE2 is PE1, and the VPNv4 route on PE2 can recurse to the BGP LSP properly.

Similar to the preceding process, if PE2 also attempts to advertise a VPNv4 route to PE1, run the **peer next-hop-invariable** command on

both RR2 and RR1 so that RR2 does not change the next hop before advertising the route to RR1 and RR1 does not change the next hop before advertising the route to PE1. In this way, the next hop of the route received by PE1 is PE2, and the VPNv4 route on PE1 can recurse to the BGP LSP properly.

e. Run **commit**

The configuration is committed.

- Configure route-policy-based next hop recursion.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **nexthop recursive-lookup { route-policy route-policy-name | route-filter route-filter-name }**

Route-policy-based next hop recursion is configured.

Route-policy-based next hop recursion helps flexibly control the recursion result based on specific conditions. If a route does not match the specified route-policy, the route fails to recurse.

e. Run **commit**

The configuration is committed.

- Prevent the device from changing the next hop address of a route to ensure that traffic is transmitted along the optimal route when the device advertises the route to a peer in the following scenarios:

- The route is learned from a directly connected peer and is to be advertised to a directly connected EBGP peer, the original next hop of the route resides on the same network segment as the local interface that is used to establish the BGP peer relationship with the EBGP peer, and all directly connected interfaces are broadcast interfaces.

- The route is locally imported and is to be advertised to a directly connected IBGP or EBGP peer, the next hop to which the route recurses resides on the same network segment as the local interface that is used to establish the BGP peer relationship with the IBGP or EBGP peer, and all directly connected interfaces are broadcast interfaces.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **nexthop third-party**

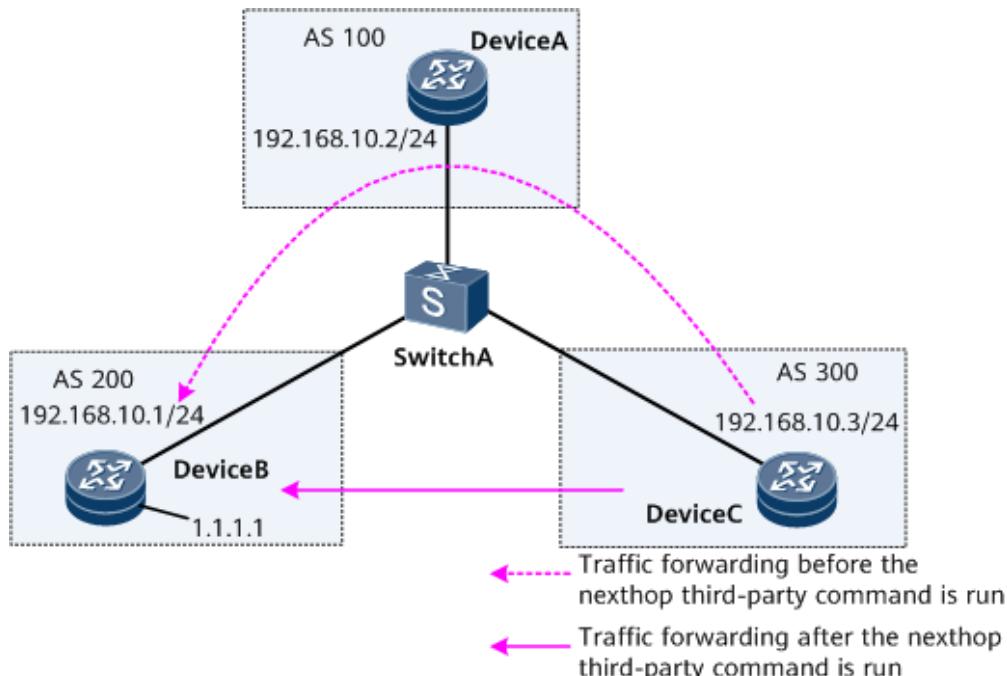
The device is prevented from changing the next hop address of a route when the device advertises the route to a peer in the specified scenarios.

e. Run **commit**

The configuration is committed.

On the network shown in [Figure 1-383](#), DeviceA establishes EBGP peer relationships with DeviceB and DeviceC, but DeviceB and DeviceC do not establish a peer relationship with each other.

Figure 1-383 Network diagram of Layer 2 transparent transmission



Assume that DeviceA receives an EBGP route 1.1.1.1 from DeviceB. If the **nexthop third-party** command is not run, the next hop address of the route is changed as follows:

- a. Before DeviceB advertises the route 1.1.1.1 to DeviceA, DeviceB changes the next hop to 192.168.10.1.
 - b. Before DeviceA advertises the route 1.1.1.1 to DeviceC, DeviceA changes the next hop to 192.168.10.2.

When traffic is transmitted from DeviceC to DeviceB, the traffic passes through DeviceA.

After the **nexthop third-party** command is run on DeviceA, the next hop address of the route 1.1.1.1 is changed as follows:

- a. Before DeviceB advertises the route 1.1.1.1 to DeviceA, DeviceB changes the route next hop to 192.168.10.1.
 - b. Before DeviceA advertises the route to DeviceC, DeviceA detects that the address (192.168.10.2) of its local interface that is used to establish the BGP peer relationship with DeviceC belongs to the same network segment as the original next hop address 192.168.10.1 of the route. Therefore, DeviceA does not change the next hop address of the route. As

a result, the next hop address of the BGP route 1.1.1.1 received by DeviceC remains 192.168.10.1.

In this way, traffic from DeviceC can be directly forwarded to DeviceB, without passing through DeviceA.

----End

Setting the AS_Path Attribute

The AS_Path attribute is used to prevent routing loops and control route selection.

Procedure

- Enable a BGP device to accept the routes that contain the local AS number if the number of repetitions in each route is within a configured limit.

Generally, BGP uses AS numbers to detect routing loops. In the Hub and Spoke networking, if EBGP runs between a Hub-PE and a Hub-CE at a Hub site, a route sent from the Hub-PE to the Hub-CE carries the AS number of the Hub-PE. If the Hub-CE sends an Update message that contains the AS number of the Hub-PE to the Hub-PE, the Hub-PE will deny it.

To ensure proper route transmission in the Hub and Spoke networking, configure all the BGP peers on the path, along which the Hub-CE advertises VPN routes to the Spoke-CE, to accept the routes that contain the local AS number if the number of repetitions in each route is within the default limit (1).

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } allow-as-loop [number]**

Repetitions of the local AS number are allowed.

In most cases, a BGP device checks the AS_Path attribute of a route received from a peer. If the AS_Path carries the local AS number, the BGP device discards this route to avoid routing loops.

In some special applications, you can use this command to allow the AS_Path attribute of a route received from a peer to contain the local AS number and set the allowed repetition count of the local AS number.

e. Run **commit**

The configuration is committed.

- Enable the local device to ignore the AS_Path attribute during route selection.

a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - d. Run **bestroute as-path-ignore**
The local device is enabled to ignore the AS_Path attribute when selecting the optimal route.
 - e. Run **commit**
The configuration is committed.
- Configure a fake AS number.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **peer { ipv4-address | group-name } fake-as fake-as-value [dual-as] [prepend-global-as] [prepend-fake-as]**
A fake AS number is configured.

This command hides the actual AS number. In this case, EBGP peers in other ASs can learn only this fake AS number of the local end and consider the fake AS number as the AS number of the local peer.

NOTE

The **peer fake-as** command is applicable only to EBGP peers.

- d. Run **commit**
The configuration is committed.
- Substitute the AS numbers in the AS_Path attribute.

In a BGP/MPLS IP VPN scenario, if the ASs to which two VPN sites belong use private AS numbers, the AS numbers of the two VPN sites may be the same. If a CE in a VPN site sends a VPN route to the connected PE using EBGP and the PE then sends the route to the remote CE, the remote CE will discard the route because the AS number carried by the route is the same as the local AS number. As a result, different sites of the same VPN cannot communicate with each other. In this case, you need to run the **peer substitute-as** command on the PE to enable AS number substitution. That is, the PE replaces the AS number of the VPN site where the CE resides in the received VPN route with the local AS number. This prevents the remote CE from discarding routes due to duplicate AS numbers.

In a BGP public network scenario, when two devices with the same AS number learn a BGP route from each other through the same EBGP peer, the route may be discarded because the AS_Path attribute contains duplicate AS numbers. To prevent this problem, run the **peer substitute-as** command on the EBGP peer shared by the two devices to enable AS number substitution.

NOTICE

Exercise caution when running the **peer substitute-as** command because improper use of the command may cause routing loops.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family { vpn-instance vpn-instance-name | unicast }**

The BGP-VPN instance IPv4 address family or BGP-IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } substitute-as**

AS number substitution is enabled on the device.

e. Run **commit**

The configuration is committed.

● Configure the AS_Path attribute to carry only the public AS number.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } public-as-only [force] [replace] [include-peer-as] [limited] [replace] [include-peer-as]]**

The AS_Path attribute in the BGP Update messages to be sent is configured to carry only public AS numbers. You can also run the **peer { ipv4-address | group-name } public-as-only import [force]** command to configure the AS_Path attribute not to carry private AS numbers in accepted BGP Update messages.

Generally, AS numbers range from 1 to 4294967295, including public AS numbers, private AS numbers, and reserved AS numbers. Private AS numbers range from 64512 to 65534 and from 4200000000 to 4294967294 (or from 64086.59904 to 65535.65534). The AS numbers 65535 and 4294967295 are reserved for special applications. Other AS numbers are public AS numbers.

NOTE

If the 4-byte private AS number capability is disabled using the **private-4-byte-as disable** command, private AS numbers range from 64512 to 65534; the AS number 65535 is reserved, and others are public AS numbers.

Public AS numbers can be used on the Internet, and are assigned and managed by the Internet Assigned Number Authority (IANA). Private AS

numbers cannot be advertised to the Internet, and are used only within ASs.

Generally, BGP routes to be advertised to peers carry either public or private AS numbers, or both. In certain cases, private AS numbers do not need to be advertised. In this case, you can run this command to configure the AS_Path attribute to carry only public AS numbers.

- e. Run **commit**

The configuration is committed.

- Set the maximum number of AS numbers in the AS_Path attribute.

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **as-path-limit limit**

The maximum number of AS numbers allowed in the AS_Path attribute is set.

After the **as-path-limit** command is run, a device checks whether the number of AS numbers in the AS_Path attribute of a received route exceeds the maximum value. If the number of AS numbers exceeds the maximum value, the device discards the route. Therefore, if the maximum number of AS numbers allowed in the AS_Path attribute is set to an excessively small value, routes may be discarded.

- d. Run **commit**

The configuration is committed.

- Disable the BGP device from checking the first AS number contained in the AS_Path attribute of each Update message received from an EBGP peer.

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **undo check-first-as**

The BGP device is disabled from checking the first AS number in the AS_Path attribute that is carried in each Update message received from an EBGP peer.

NOTICE

Exercise caution when running the **undo check-first-as** command because use of this command may cause routing loops.

- d. Run **commit**

The configuration is committed.

After the configuration is complete, run the **refresh bgp** command to check the received routes again.

- Enable the device to check or disable the device from checking the first AS number in the AS_Path attribute contained in the Update messages received from a specified EBGP peer or peer group.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **peer { group-name | ipv4-address } check-first-as { enable | disable }**

The device is enabled to check or disabled from checking the first AS number in the AS_Path attribute contained in each Update message received from a specified EBGP peer or peer group.

If the **peer check-first-as enable** command is run, BGP checks whether the first AS number in the AS_Path attribute of each Update message received from the specified EBGP peer or peer group is the same as the number of the AS where the EBGP peer or peer group resides. The two AS numbers must be the same. If the AS numbers are different, the Update message is rejected. If the **peer check-first-as disable** command is run, BGP accepts Update messages received from the specified EBGP peer or peer group, regardless of whether the first AS number in the AS_Path attribute of the messages is the same as the number of the AS where the EBGP peer or peer group resides. If the undo command is run, the related configuration for the specified EBGP peer or peer group is deleted, and the default configuration is used.

The check on the first AS number in the AS_Path attribute of each received Update message can be configured for a specified EBGP peer, the peer group that the EBGP peer belongs to, or the entire BGP process. The configuration takes effect in descending order of priority as follows: EBGP peer > peer group > entire BGP process.

d. Run **commit**

The configuration is committed.

After the configuration is complete, run the **refresh bgp** command to check the received routes again.

----End

Configuring AIGP Attributes for Routes

The Accumulated Interior Gateway Protocol (AIGP) attribute allows devices in an AIGP domain to use the optimal routes to forward data.

Context

Generally, a set of ASs managed by the same administrative department is called an AIGP administrative domain, or AIGP domain for short.

IGPs running in an administrative domain assign a metric to each link and select the path with the smallest metric. BGP, designed to provide routing over a large number of independent administrative domains, does not select paths based on metrics. If a single administrative domain (AIGP domain) consists of several BGP networks, it is desirable for BGP to select paths based on metrics, just as an IGP does.

After the AIGP attribute is configured in an AIGP domain, BGP selects optimal routes based on route metrics, just as an IGP does. This ensures that all devices in the AIGP domain forward data along the optimal paths.

Procedure

Step 1 Enable AIGP capability for a specified peer or peer group.

1. Run **system-view**

The system view is displayed.

2. Run **bgp as-number**

The BGP view is displayed.

3. Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

4. Run **peer { group-name | ipv4-address } aigp**

The AIGP capability is enabled for the specified peer or peer group.

BGP allows you to configure AIGP for a single peer or peer group. The configuration on a peer takes precedence over the configuration on a peer group. If a BGP peer without the AIGP capability joins a BGP peer group that has the AIGP capability, the BGP peer inherits the AIGP capability of the BGP peer group. After a BGP peer inherits the AIGP capability of a BGP peer group, you can still run the **undo peer aigp** command to delete the AIGP configuration.

5. Run **commit**

The configuration is committed.

Step 2 (Optional) Allow public network routes to participate in route selection using the AIGP attribute of the corresponding BGP LSP.

1. Run **system-view**

The system view is displayed.

2. Run **bgp as-number**

The BGP view is displayed.

3. Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

4. Run **bestroute nexthop-resolved aigp**

Public network routes are allowed to participate in route selection using the AIGP attribute of the corresponding BGP LSP.

5. Run **commit**

The configuration is committed.

----End

Configuring Attr_Set Attribute Encapsulation or Parsing

Configuring the Attr_Set attribute for BGP routes ensures that CE route attributes are transparently transmitted over the backbone network.

Context

On BGP MPLS/VPN networks, EBGP peer relationships are established between PEs and CEs in most cases. Attributes of the routes advertised by CEs are modified during transmission over the intermediate backbone network, or the attributes affect the backbone network. In this case, BGP has been extended to allow the intermediate backbone network to transparently transmit the routes advertised by CEs. After receiving a route from a CE, the local PE encapsulates the attributes of the route in the Attr_Set attribute and then sends the route to the remote PE. Upon receipt of the route, the remote PE parses the Attr_Set attribute. This process ensures that the route attributes are transparently transmitted over the backbone network.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family vpn-instance vpn-instance-name** command to enter the BGP-VPN instance IPv4 address family view or run the **ipv6-family vpn-instance vpn-instance-name** command to enter the BGP-VPN instance IPv6 address family view.

Step 4 Run **attr-set { both | send | receive }**

The device is configured to encapsulate the Attr_Set attribute when sending VPN routes, parse the Attr_Set attribute when receiving VPN routes, or encapsulate the Attr_Set attribute when sending VPN routes and parse the Attr_Set attribute when receiving VPN routes.

Step 5 (Optional) Run **attr-set as-path-check enable**

AS_Path check is enabled for loop detection.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the BGP Route Attribute Configuration

After configuring BGP route selection, verify information about route attributes.

Prerequisites

BGP route attributes have been configured.

Procedure

- Run the **display bgp routing-table different-origin-as** command to check routes with different source ASs but the same destination address.
- Run the **display bgp routing-table regular-expression as-regular-expression** command to check routes matching the AS regular expression.
- Run the **display bgp routing-table [network] [mask | mask-length] [longer-prefixes]** command to check information about the BGP routing table.
- Run the **display bgp routing-table community [community-number | aa:nn] &<1-13> [internet | no-advertise | no-export | no-export-subconfed] * [whole-match]** command to check routes matching a specified BGP community attribute.
- Run the **display bgp routing-table community-filter { { community-filter-name | basic-community-filter-number } [whole-match] | advanced-community-filter-number }** command to check the routes matching a specified BGP community filter.

----End

1.1.9.2.6 Configuring BGP Routing Policies

BGP is used to transmit routing information between ASs, and its routing policies can be used to flexibly control route advertisement and acceptance, which directly affect traffic forwarding.

Usage Scenario

A large number of routes typically exist in a BGP routing table, and transmitting such extensive routing information brings heavy burden to a device. In order to address this problem, it is necessary to filter those routes to be advertised. You can configure a device to advertise only necessary routes or those that its peers require. Multiple routes to the same destination may exist and traverse different ASs. To direct traffic to specific ASs, routes to be advertised also need to be filtered.

A BGP device may receive multiple routes to the same destination network from different peers. To control the network traffic forwarding path, BGP needs to filter the received routes. In addition, due to potential service attacks, the BGP device may receive an excessively large number of routes from its peers, consuming many resources on the device. Regardless of whether malicious attacks or incorrect configurations result in the reception of numerous BGP routes, the administrator must limit the device resources to be consumed based on the network planning and device capacity.

Filters can be used to filter the routes to be advertised and received by BGP. BGP can filter the routes to be advertised by a peer or peer group, the routes that are received globally, or the routes received from a peer or peer group. If multiple filter policies are configured, BGP advertises or accepts only the routes that match all the filter policies.

Pre-configuration Tasks

Before configuring BGP to advertise routes, complete the following tasks:

- [Configure basic BGP functions.](#)

Configuring BGP Filters

BGP filters can be used to flexibly filter routes to be advertised.

Context

BGP uses the following types of filters:

- [Access Control List \(ACL\)](#)
- [IP prefix list](#)
- [AS_Path](#)
- [Community](#)
- [Large-community](#)
- [Extended community](#)
- [Route-policy](#)
- [IP address list](#)

Procedure

- Configure an ACL.

An ACL is a series of sequential rules composed of permit and deny clauses. These rules specify source addresses, destination addresses, port numbers, and other information of data packets. ACL rules are used to classify data packets, and after the rules are applied to an interface on the router, the router uses the rules to permit or deny data packets.

For details on ACL configurations, see HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - IP Services.

An ACL can be used as a matching condition of a route-policy or directly used in the following commands:

- **`filter-policy { acl-number | acl-name acl-name } export [direct | isis process-id | ospf process-id | rip process-id | static]`**
- **`peer { group-name | ipv4-address } filter-policy { acl-number | acl-name acl-name } export`**
- Configure an IP prefix list.

An IP prefix list is a type of filter used to filter routes based on destination addresses, and is identified by its name. An IP prefix list can be used flexibly to implement precise filtering. For example, it can be used to filter one route or routes to a network segment. If a large number of routes with different prefixes need to be filtered, configuring an IP prefix list to filter the routes is very complex.

An IP prefix list can be used as a matching condition of a route-policy or directly used in the following commands:

- **filter-policy ip-prefix ip-prefix-name export** [**direct** | **isis process-id** | **ospf process-id** | **rip process-id** | **static**]
- **peer { group-name | ipv4-address } ip-prefix ip-prefix-name export**
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **ip ip-prefix ip-prefix-name [index index-number] matchMode ipv4-address masklen [match-network] [greater-equal greater-equal-value] [less-equal less-equal-value]** command to configure an IPv4 prefix list.

The mask length range can be expressed as *mask-length* <= *greater-equal-value* <= *less-equal-value* <= 32. If only **greater-equal** is specified, the prefix range is [*greater-equal-value*, 32]. If only **less-equal** is specified, the prefix range is [*mask-length*, *less-equal-value*].

An IPv4 prefix list is identified by its name, and each list contains one or multiple entries. Each entry is identified by an index, and can uniquely specify a matching range in the form of a network prefix. An IPv4 prefix list named **abcd** is used as an example.

```
#  
ip ip-prefix abcd index 10 permit 1.0.0.0 8  
ip ip-prefix abcd index 20 permit 10.0.0.0 8
```

During route matching, the system checks the entries identified by indexes in ascending order. If a route matches an entry, it is no longer matched against the next entry.

The IP prefix list on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X denies all unmatched routes by default. If all entries are in **deny** mode, all routes will be denied by the IP prefix list. In this case, after multiple entries are specified in **deny** mode, define an entry **permit 0.0.0.0 0 less-equal 32** to permit all the other IPv4 routes.

NOTE

If more than one entry is defined in a prefix list, at least one of them must be set to **permit** mode.

- c. Run the **commit** command to commit the configuration.
- Configure an AS_Path filter.

An AS_Path filter is used to filter BGP routes based on the AS_Path attributes they carry. If you do not want traffic to traverse some ASs, you can configure an AS_Path filter to filter out the routes whose AS_Path attributes contain the corresponding AS numbers. In addition, configuring an ACL or an IP prefix list to filter BGP routes may be complicated (as multiple ACLs or IP prefix lists need to be defined) and complicate maintenance of new routes. In this case, you can configure an AS_Path filter.

NOTE

If the AS_Path information of summary routes is lost, an AS_Path filter can no longer filter them. However, it can still filter specific routes, which carry AS_Path information.

An AS_Path filter can be used as a filtering condition of a route-policy or be directly used in the **peer as-path-filter** command.

- a. Run the **system-view** command to enter the system view.
- b. Run the **ip as-path-filter { as-path-filter-number| as-path-filter-name }** [**index index-number**] **matchMode regular-expression** command to configure an AS_Path filter.

AS_Path filters use regular expressions to define matching rules. A regular expression consists of metacharacters and values. For details, see [Table 1-124](#).

- Metacharacter: defines matching rules.
- Value: defines a matching object.

Table 1-124 Description of metacharacters

Metacharacter	Function	Example
.	Matches any single character.	.* matches any string in an AS_Path and is used to match all routes.
^	Indicates the beginning of a matched string.	 ^65 matches strings beginning with 65. <ul style="list-style-type: none">● Examples of matched strings: 65, 651, 6501, and 65001● Examples of unmatched strings: 165, 1650, 6650, and 60065
\$	Indicates the end of a matched string.	65\$ matches strings ending with 65. <ul style="list-style-type: none">● Examples of matched strings: 65, 165, 1065, 10065, and 60065● Examples of unmatched strings: 651, 1650, 6650, 60650, and 65001 ^65\$ matches AS_Path 65. NOTE ^\$ matches an empty string (empty AS_Path) and is usually used to match routes in the local AS.

Metacharacter	Function	Example
-	Matches a sign, such as a comma (,), left brace ({}, right brace {}, left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning of a regular expression with the same function as the caret sign (^) or at the end of a regular expression with the same function as the dollar sign (\$).	<ul style="list-style-type: none">• ^65001_ matches the AS_Paths that begin with 65001 followed by a sign. Specifically, ^65001_ matches AS_Paths with 65001 as the leftmost AS number (the number of the last AS through which a route passes), that is, the routes sent by peers in AS 65001.• _65001_ matches the strings (AS_Paths) that contain 65001, that is, the routes that pass through AS 65001.• _65001\$ matches the AS_Paths that end with a sign followed by 65001. Specifically, _65001\$ matches AS_Paths with 65001 as the rightmost AS number (the number of the first AS through which a route passes), that is, the routes that originate in AS 65001.

Metacharacter	Function	Example
\	Defines an escape character, which is used to mark the next character (common or special) as a common character.	An AS_Confed_Sequence contains parentheses. The parentheses in regular expressions provide special functions. To match such special characters by removing their special meanings, you can use the backslash (\). For example: <ul style="list-style-type: none">• \(65002_ matches the AS_Confed_Sequences that begin with (65002 followed by a sign. Specifically, \(65002_ matches AS_Confed_Sequences with 65002 as the leftmost AS number (the number of the last AS through which a route passes), that is, the routes sent by peers in AS 65002.• \(.*_65003_.*) matches the AS_Confed_Sequence that contains AS number 65003, that is, the routes that pass through AS 65003 in a confederation.• _65004_) matches a character string that ends with 65004) and is preceded by a sign. Specifically, _65004_) matches AS_Confed_Sequences with the rightmost AS number (origin AS number), that is, the routes originating in AS 65004 in a confederation and the routes directly advertised by AS 65004 in the confederation. _65004_) provides the same function as 65004). Similarly, backslashes (\) can be used to remove the special meanings of the left bracket ([) and right bracket (]) used

Metacharacter	Function	Example
		in an AS_Confederation_Set and the left brace {} and right brace {} used in an AS_Set.
*	Matches the strings in which the preceding character occurs zero or multiple times.	<p>65* matches the AS_Paths that begin with 6 and contain zero or multiple 5s.</p> <ul style="list-style-type: none"> • Examples of matched strings: 6, 65, 655, 6559, 65259, and 65529 • Examples of unmatched strings: 5, 56, 556, 5669, 55269, and 56259
+	Matches the strings in which the preceding character occurs one or more times.	<p>65+ matches the AS_Paths that begin with 6 and contain one or multiple 5s.</p> <ul style="list-style-type: none"> • Examples of matched strings: 65, 655, 6559, 65259, and 65529 • Examples of unmatched strings: 56, 556, 5669, 55269, and 56259
?	Matches the strings in which the preceding character occurs zero or one time.	<p>65? matches the AS_Paths that begin with 6 and contain zero or one 5.</p> <ul style="list-style-type: none"> • Examples of matched strings: 6 and 65 • Examples of unmatched strings: 655, 6559, and 65529
()	Matches a sub-regular expression within the parentheses, and obtains the matching result. The parentheses can also be empty.	100(200)+ matches 100200, 100200200, and so on.
x y	Matches x or y.	100 65002 65003 matches 100, 65002, or 65003.
[xyz]	Matches any character in the regular expression.	[896] matches 8, 9, or 6.

Metacharacter	Function	Example
[^xyz]	Matches any character that is not contained in the regular expression.	[^896] matches any character, except 8, 9, and 6.
[a-z]	Matches any character within the specified range.	[2-4] matches 2, 3, and 4; [0-9] matches any digits from 0 to 9. NOTE The values in the square brackets ([]) must be digits from 0 to 9. For example, to match a number ranging from 735 to 907, use the regular expression of (73[5-9] 7[4-9][0-9] 8[0-9][0-9] 90[0-7]).
[^a-z]	Matches any character beyond the specified range.	[^2-4] matches characters other than 2, 3, and 4. [^0-9] matches characters other than digits 0 through 9.

You can define multiple rules (permit or deny) for the same filter. During the matching, the relationship between these rules is OR. If a route meets one of the matching rules, it matches this AS_Path filter.

NOTE

For details on a regular expression, see the HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - Basic Configurations.

- c. Run the **commit** command to commit the configuration.
- Configure a community filter.

A BGP community attribute is used to identify a group of routes with the same properties. Routes can be classified through the community attribute, which facilitates route management.

In actual application, some AS internal routes may not need to be advertised to other ASs, while some AS external routes need to be advertised to other ASs. Both internal and external routes may have different prefixes (making an IP prefix list unsuitable), and may come from different ASs (making an AS_Path filter unsuitable). In this case, you can set a community value for the AS internal routes and another community value for the AS external routes on an AS edge device to control and filter routes.

- a. Run the **system-view** command to enter the system view.
- b. Run the **ip community-filter adv-comm-filter-num [index index-number] matchMode regular-expression** command to configure a community filter.

- To configure a standard community filter, run the **ip community-filter basic basCfName [index index-val] matchMode [cmntyStr cmntyNum | internet [strict-match] | no-advertise | no-export | no-export-subconfed] &<1-20>** or **ip community-filter cfIndex [index index-val] matchMode [cmntyStr | cmntyNum | internet [strict-match] | no-advertise | no-export | no-export-subconfed] &<1-20>** command.
- To configure an advanced community filter, run the **ip community-filter { advanced comm-filter-name | advanced adv-comm-filter-num } [index index-number] { permit | deny } regular-expression** command.
 - c. Run the **commit** command to commit the configuration.
- Configure a Large-community filter.

The Large-community attribute can completely represent a 2-byte or 4-byte AS number, and has two 4-byte LocalData fields, allowing the administrator to flexibly use route-policies. As an enhancement to community attributes, Large-community can be used together with community attributes.

 - a. Run the **system-view** command to enter the system view.
 - b. Configure a Large-community filter.
 - To configure a basic Large-community filter, run the **ip large-community-filter basic large-comm-filter-name [index index-number] { permit | deny } { aa:bb:cc } &<1-16>** command.
 - To configure an advanced Large-community filter, run the **ip large-community-filter advanced large-comm-filter-name [index index-number] { permit | deny } regular-expression** command.
 - c. Run the **commit** command to commit the configuration.

● Configure an extended community filter.

Similar to a community filter, an extended community filter is mainly used to filter VPN routes.

- a. Run the **system-view** command to enter the system view.
- b. Configure the following extended community filters as needed.

To configure a VPN-Target extended community filter:

- To configure a basic VPN-Target extended community filter, run the **ip extcommunity-filter { basic-extcomm-filter-num | basic basic-extcomm-filter-name } [index index-number] { deny | permit } { rt extCmntyStr } &<1-16>** command.
- To configure an advanced VPN-Target extended community filter, run the **ip extcommunity-filter { advanced-extcomm-filter-num | advanced advanced-extcomm-filter-name } [index index-number] { deny | permit } regular-expression** command.

To configure an SoO extended community filter:

- To configure a basic SoO extended community filter, run the **ip extcommunity-list soo basic *basic-extcomm-filter-name* [index *index-number*] { permit | deny } { site-of-origin }** &<1-16> command.
- To configure an advanced SoO extended community filter, run the **ip extcommunity-list soo advanced *advanced-extcomm-filter-name* [index *index-number*] { permit | deny } *regular-expression*** command.

Multiple entries can be defined in one extended community filter identified either by a name or number. The relationship between the entries is "OR". This means that if a route matches one of the rules, the route matches the filter.

- c. Run the **commit** command to commit the configuration.
- Configure a route-policy.

A route-policy is used to match routes or route attributes, and to change route attributes when the matching rules are met. As the preceding filters can be used as matching conditions of a route-policy, the route-policy offers powerful functions and can be used flexibly.

- a. Run the **system-view** command to enter the system view.
- b. Run the **route-policy *route-policy-name* matchMode node *node*** command to configure a route-policy with a node and enter the route-policy view.

A route-policy may consist of multiple nodes. For example, the **route-policy *route-policy-example* permit node 10** command defines node 10 and the **route-policy *route-policy-example* deny node 20** command defines node 20. Both nodes belong to the route-policy named **route-policy-example**. The relationship between the nodes of a route-policy is OR. There are two situations:

- If a route matches one node, it matches the route-policy and will not be matched against the next node. In the preceding example, if a route matches the node defined using the **route-policy *route-policy-example* permit node 10** command, the route will not be matched against the node defined using the **route-policy *route-policy-example* deny node 20** command.
- If a route does not match any node, the route fails to match the route-policy.

When a route-policy is used to filter routes, the routes are first matched against the node with the smallest node ID. For example, if two nodes are configured using the **route-policy *route-policy-example* permit node 10** and **route-policy *route-policy-example* deny node 20** commands, routes are first matched against the node configured using the **route-policy *route-policy-example* permit node 10** command.

 NOTE

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, all unmatched routes are denied by the route-policy by default. If more than one node is defined in a route-policy, at least one of them must be in **permit** mode.

- c. (Optional) Configure **if-match** clauses for the current route-policy node.

if-match clauses are used to filter routes. If no **if-match** clause is specified for a node, all routes will match the node.

- To configure an ACL-based matching rule, run the **if-match acl { acl-number | acl-name }** command.
- To match routes against an IP prefix list, run the **if-match ip-prefix ip-prefix-name** command.

 NOTE

The **if-match acl** and **if-match ip-prefix** commands cannot be both used in the same route-policy node. If they are both configured in the same route-policy node, the latter configuration overrides the previous one.

- To match the AS_Path attribute of BGP routes, run the **if-match as-path-filter apflIndex &<1-16>** command.
- To match the community attribute of BGP routes, run either of the following commands:
 - **if-match community-filter { basic-comm-filter-num [whole-match] | adv-comm-filter-num [sort-match] }* &<1-16>**
 - **if-match community-filter comm-filter-name [whole-match | sort-match]**
- To match the Large-community attribute of BGP routes, run the **if-match large-community-filter large-comm-filter-name [whole-match]** command.
- To match the VPN target extended community attribute of BGP routes, run the **if-match extcommunity-filter { { basic-extcomm-filter-num [matches-all | whole-match] | adv-extcomm-filter-num } &<1-16> | extcomm-filter-name [matches-all | whole-match] }** command.
- To match the SoO extended community attribute of BGP routes, run the **if-match extcommunity-list soo extcomm-filter-name** command.

The operations in Step 3 can be performed in any order. A node may have multiple **if-match** clauses or none at all.

 NOTE

The relationship between **if-match** clauses in a route-policy node is AND, meaning that a route must match all matching conditions before the action defined by an **apply** clause is taken on this route. For example, if two **if-match** clauses (**if-match acl 2003** and **if-match as-path-filter 100**) are defined in the route-policy node specified using the **route-policy route-policy-example permit node 10** command, a route is considered to match node 10 only when it matches both **if-match** clauses.

- d. (Optional) Configure **apply** clauses for the current route-policy node.

apply clauses can be used to set attributes for the routes matching the **if-match** clauses. If this step is not performed, the attributes of the routes matching the **if-match** clause remain unchanged.

- To replace the AS numbers in the AS_Path attribute of BGP routes or add the specified AS number to the AS_Path attribute, run the **apply as-path { as-path-value } &<1-128>{ additive | overwrite | delete }** or **apply as-path asValues { additive | overwrite | delete }** command.
- To delete a specified BGP community attribute, run the **apply comm-filter delete** command.

 NOTE

The **apply comm-filter delete** command deletes a specified community attribute from matched routes based on the referenced community filter. To delete multiple community attributes through one community filter, you need to run the **ip community-filter** command multiple times to configure multiple indexes for the filter, with each index corresponding to only one community attribute. If multiple community attributes are specified in the same index of the same community filter, none of them can be deleted in this case. For detailed examples, see the HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Command Reference.

- To delete community attributes from matched BGP routes, run the **apply community none** command.
- To set the community attribute of matched BGP routes, run the **apply community { cmntyValue | cmntyNum | internet | no-advertise | no-export | no-export-subconfed } &<1-32> [additive]** or **apply community community-list community-list-name** command.

 NOTE

A BGP community list must have been configured using the **ip community-list** command and community attributes must have been configured for the list using the **community** command before you run the **apply community community-list community-list-name** command.

- To delete the Large-community attribute of BGP routes, run the **apply large-community none** command.
- To set the Large-community attribute for BGP routes, run the **apply large-community { aa:bb:cc } &<1-16> { additive | overwrite | delete }** or **apply large-community-list large-community-list-name { additive | overwrite | delete }** command.

 NOTE

Before running the **apply large-community-list** *large-community-list-name* command to set the BGP Large-community attribute, run the **ip large-community-list** command to configure a BGP Large-community list and run the **large-community** command to configure values for the Large-community list.

- To set the BGP VPN target extended community attribute, run the **apply extcommunity { rt extCmntyValue } &<1-16> [additive]** command.
- To set the BGP SoO extended community attribute, run the **apply extcommunity soo { site-of-origin } &<1-16> additive** command.
- To set the bandwidth extended community attribute, run the **apply extcommunity bandwidth { extCmntyString | none } or apply extcommunity bandwidth aggregate [limit bandwidth-value]** command.
- To set the Local_Pref attribute for matched BGP routes, run the **apply local-preference [+ | -] preference** command.
- To set the Origin attribute for matched BGP routes, run the **apply origin { egp { egpVal } | igrp | incomplete }** command.
- To set a preferred value for matched BGP routes, run the **apply preferred-value** *preferred-value* command.
- To set dampening parameters for EBGP routes, run the **apply dampening half-life-reach reuse suppress ceiling** command.

The operations in Step 4 can be performed in any order. A node may have multiple **apply** clauses or no **apply** clause.

- e. Run the **commit** command to commit the configuration.
- Configure an IP address list.

Before configuring a conditional BGP route advertisement policy, you need to create an IP address list.

- a. Run the **system-view** command to enter the system view.
- b. Run the **filter-list ip-prefix** *name* command to create an IP address list and enter the ip-prefix-list view.
- c. Run the **prefix** *address maskLen* command to configure an IP address and mask for the IP address list.
- d. Run the **commit** command to commit the configuration.

----End

Applying a Policy to BGP Route Advertisement

After a route advertisement policy is configured on a device, the device advertises only the routes that match the policy to its BGP peers.

Procedure

- Configure BGP to filter the routes to be advertised globally.

You can configure the device to filter the routes to be advertised. Perform the following steps on a BGP router:

- Run **system-view**

The system view is displayed.

- Run **bgp as-number**

The BGP view is displayed.

- Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

- Configure BGP to filter routes to be advertised globally.

■ To filter routes based on a basic ACL, perform the following steps:

- Run the **filter-policy { acl-number | acl-name acl-name } export [direct | isis process-id | ospf process-id | rip process-id | static]** command to filter the routes to be advertised.
- Run the **quit** command to return to the BGP view.
- Run the **quit** command to return to the system view.
- Run the **acl { name basic-acl-name { basic | [basic] number basic-acl-number } } [match-order { config | auto }]** command to enter the ACL view.
- Run the **rule [rule-id] [name rule-name] { deny | permit }** command to configure a rule for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the

unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- To filter routes based on an IP prefix list, run the **filter-policy ip-prefix ip-prefix-name export [direct | isis process-id | ospf process-id | rip process-id | static]** command.

If a protocol type is specified, only the routes discovered by the specified routing protocol match the filtering condition. If no protocol type is specified, all BGP routes to be advertised match the filtering condition, including those imported using the **import-route (BGP)** command and local routes imported using the **network (BGP)** command.

NOTE

If an ACL is specified in the **filter-policy** command and no VPN instance is specified in ACL rules, BGP filters both public network and VPN routes in all address families. If a VPN instance is specified in an ACL rule, only data traffic from the VPN instance matches the filtering condition, and no routes are filtered.

e. Run **commit**

The configuration is committed.

- Configure BGP to filter the routes to be advertised to a specified peer or peer group.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

- d. Perform any of the following operations to filter the routes to be advertised to a specified peer or peer group:

- To filter routes based on a basic ACL, perform the following steps:

- 1) Run the **peer { ipv4-address | group-name } filter-policy { acl-number | acl-name acl-name } export** command to configure the device to filter the routes to be advertised to a specified peer or peer group.
- 2) Run the **quit** command to return to the BGP view.
- 3) Run the **quit** command to return to the system view.
- 4) Run the **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]** command to enter the ACL view.

- 5) Run the **rule** [*rule-id*] [**name** *rule-name*] {**deny** | **permit**} command to configure a rule for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:

If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a small ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the routes that are not to be accepted or advertised.

- To filter routes based on an IP prefix list, run the **peer** {*ipv4-address* | *group-name*} **ip-prefix** *ip-prefix-name* **export** command.
- To filter routes based on an AS_Path filter, run the **peer** {*ipv4-address* | *group-name*} **as-path-filter** {*number* | *name*} **export** command.
- To filter routes based on a route-policy, run the **peer** {*ipv4-address* | *group-name*} **route-policy** *route-policy-name* **export** command.

 **NOTE**

A route-policy specified in the **peer route-policy export** command does not support interface-based matching rules (configured using the **if-match interface** command).

- To filter routes based on an IP address list, run the **peer** {*ipv4-address* | *group-name*} **advertise dependent-filter** *dependent-filter-*

list outDependType [condition-filter condition-filter-list | condition-ip-filter ip-prefix-name] command.

 NOTE

If BGP route status changes after a filtering policy that is based on an IP address list is enabled, the routes are re-advertised based on the conditional advertisement policy 10 seconds later by default. To set the delay in advertising the routes, run the **timer dependent-advertise-delay delay-time** command.

The export routing policy applied to a peer in a peer group can be different from that applied to the peer group. Specifically, a unique policy can be used to filter the routes to be advertised to each peer in the peer group.

e. Run **commit**

The configuration is committed.

----End

Applying a Policy to BGP Route Acceptance

After an import policy is configured, only the routes that match the import policy are accepted.

Procedure

- Configure BGP to filter the routes to be received globally.

You can configure the device to filter the routes to be received.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Configure BGP to filter the routes to be received globally.

▪ To filter routes based on a basic ACL, perform the following steps:

- Run the **filter-policy { acl-number | acl-name acl-name } import** command to filter the routes to be received.
- Run the **quit** command to return to the BGP view.
- Run the **quit** command to return to the system view.
- Run the **acl { name basic-acl-name { basic | [basic] number basic-acl-number } } [match-order { config | auto }]** command to enter the ACL view.
- Run the **rule [rule-id] [name rule-name] { deny | permit }** command to configure a rule for the ACL.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- To filter routes based on an IP prefix list, run the **filter-policy ip-prefix ip-prefix-name import** command.

NOTE

If an ACL is specified in the **filter-policy** command and no VPN instance is specified in ACL rules, BGP filters both public network and VPN routes in all address families. If a VPN instance is specified in an ACL rule, only data traffic from the VPN instance matches the filtering condition, and no routes are filtered.

e. Run **commit**

The configuration is committed.

- Configure BGP to filter the routes to be received from a specified peer or peer group.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Perform any of the following operations to filter the routes to be received from a specified peer or peer group:

- To filter routes based on a basic ACL, perform the following steps:
 - 1) Run the **peer { ipv4-address | group-name } filter-policy { acl-number | acl-name acl-name } import** command to configure the device to filter the routes to be received from a specified peer or peer group.
 - 2) Run the **quit** command to return to the BGP view.
 - 3) Run the **quit** command to return to the system view.
 - 4) Run the **acl { name basic-acl-name { basic | [basic] number basic-acl-number } | [number] basic-acl-number } [match-order { config | auto }]** command to enter the ACL view.
 - 5) Run the **rule [rule-id] [name rule-name] { deny | permit } [fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first }] | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name] * command to configure a rule for the ACL.**

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number

in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- To filter routes based on an IP prefix list, run the **peer { ipv4-address | group-name } ip-prefix ip-prefix-name import** command.
- To filter routes based on an AS_Path filter, run the **peer { ipv4-address | group-name } as-path-filter { number | name } import** command.
- To filter routes based on a route-policy, run the **peer { ipv4-address | group-name } route-policy route-policy-name import** command.

 **NOTE**

A route-policy specified in the **peer route-policy import** command does not support interface-based matching rules (configured using the **if-match interface** command).

The import policy applied to a peer in a peer group can be different from that applied to the peer group. Specifically, a unique policy can be used to filter the routes to be received from each peer in the peer group.

e. Run **commit**

The configuration is committed.

- Limit the number of routes received from peers.

If a BGP router is maliciously attacked or network configuration errors occur, the router will receive a large number of routes from its peers. As a result, a large number of resources are consumed on the router. To prevent this issue, the administrator must limit the resources to be consumed during device operation based on the network planning and router capacity. BGP provides peer-specific route control to limit the number of routes received from a peer or peer group.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { group-name | ipv4-address } route-limit limit [percentage] [alert-only | idle-forever | idle-timeout minutes]**

The maximum number of routes that the device is allowed to accept from the specified peer or peer group is set.

This command provides route control based on peers. You can specify parameters as needed to control the processing if the number of the routes accepted from a peer exceeds the threshold.

- If **alert-only** is set, the device does not terminate the connection but does not accept any excess routes after the threshold is reached. In addition, the device generates an alarm and a log.

- If **idle-forever** is set, the peer relationship is interrupted and the system does not automatically attempt to re-establish the connection. In addition, the device generates an alarm and a log. In this case, the peer status is Idle in the **display bgp peer [verbose]** command output. To restore the BGP connection, run the **reset bgp** command. This parameter is not recommended.
- If **idle-timeout** is set, the peer relationship is interrupted. After the timer expires, the device tries to re-establish the peer relationship. In addition, the device generates an alarm and a log. In this case, the peer status is Idle in the **display bgp peer [verbose]** command output. To restore the BGP connection before the timer expires, run the **reset bgp** command.
- If none of the preceding parameters is specified, the peer relationship is terminated, and the device tries to re-establish the peer relationship in 30 seconds. In addition, the device generates an alarm and a log.

 NOTE

If the number of routes received by a router exceeds the limit and the **peer route-limit** command is used for the first time, the router re-establishes the peer relationship with the peer, regardless of whether **alert-only** is specified.

e. Run **commit**

The configuration is committed.

----End

Configuring BGP Soft Reset

When a BGP policy is changed, BGP can apply the new policy immediately and refresh the BGP routing table dynamically without tearing down any BGP connection.

Context

After a BGP policy is changed, you can reset the BGP connection for the new policy to take effect immediately. This, however, interrupts the BGP connection temporarily. BGP route-refresh allows the system to refresh the BGP routing table dynamically without tearing down any BGP connection when a policy is changed.

- If a BGP peer supports route-refresh, you can run the **refresh bgp** command to softly reset the BGP connection to refresh the routing table.
- If a BGP peer does not support route-refresh, you can run the **peer keep-all-routes** command to reserve all the original routes of the peer. In this manner, the routing table can be refreshed without resetting the BGP connection.

Procedure

- For BGP peers that support route-refresh:
 - a. (Optional) Enable route-refresh.
 - i. Run **system-view**

The system view is displayed.

- ii. Run **bgp as-number**

The BGP view is displayed.

- iii. Run **peer { ipv4-address | group-name } capability-advertise route-refresh**

Route-refresh is enabled.

- iv. Run **commit**

The configuration is committed.

If route-refresh is enabled on all BGP peers and the import routing policy of the local router is changed, the local router sends a Route-refresh message to its peers or peer groups. Upon receipt, the peers or peer groups re-send their routing information to the local router. This enables the local router to dynamically update its BGP routing table and apply the new without terminating any BGP connections.

- b. Configure BGP soft reset.

- i. Run the **refresh bgp [vpn-instance vpn-instance-name ipv4-family | vpng4] { all | ipv4-address | group group-name | external | internal } import** command in the user view to trigger BGP soft reset immediately.

external softly resets an EBGP connection, and **internal** softly resets an IBGP connection.

- For BGP peers that do not support route-refresh:

- Configure the device to retain all the routing updates received from a specified peer or peer group.

- i. Run **system-view**

The system view is displayed.

- ii. Run **bgp as-number**

The BGP view is displayed.

- iii. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

- iv. Run **peer { ipv4-address | group-name } keep-all-routes**

The device is configured to retain all routing updates received from the specified peer or peer group.

After this command is run, all routing updates received from the specified peer or peer group are retained, regardless of whether an import policy is used. If the local routing policy changes, the retained information can be used to regenerate BGP routes.

NOTE

This command needs to be run on both the local device and its peers. If the **peer keep-all-routes** command is run on the device for the first time, the sessions between the device and its peers are re-established.

The **peer keep-all-routes** command does not need to be run on the router that support route-refresh. If this command is run, the sessions between the router and its peers are not reestablished, but the **refresh bgp** command does not take effect on the router.

v. Run **commit**

The configuration is committed.

----End

Configuring BGP Message Extension

With the enhancement of BGP capabilities, a BGP session needs to negotiate multiple capabilities and use BGP messages to carry information about the routes and route attributes to be advertised. However, the length of BGP messages is limited. You can enable BGP message extension to remove the limit on the length of BGP messages.

Procedure

Step 1 Run the **system-view** command to enter the system view.

Step 2 Run the **bgp *as-number*** command to enter the BGP view.

Step 3 Run the **peer { peerIpv4Addr | peerGroupName } extended-open-message** command to enable the BGP Open message extension function so that the length of an Open message can be greater than 255 bytes.

Step 4 Run the **commit** command to commit the configuration.

----End

Verifying the Configuration

After the configurations for controlling BGP route advertisement and acceptance are configured, you can view information about the configured filters, routes that match the specified filter, routes advertised to peers, and the imported routes that match the specified filter.

Prerequisites

All configurations for controlling BGP route advertisement and acceptance have been completed.

Procedure

- Run the **display ip as-path-filter [*as-path-filter-number* | *as-path-filter-name*]** command to check information about a configured AS_Path filter.
- Run the **display ip community-filter [*basic-comm-filter-num* | *adv-comm-filter-num* | *comm-filter-name*]** command to check information about a configured community filter.
- Run the **display ip extcommunity-filter [*basic-extcomm-filter-num* | *advanced-extcomm-filter-num* | *extcomm-filter-name*]** command to check the VPN-Target extended community filter information.
- Run the **display ip extcommunity-list *soo* [*extcomm-filter-name*]** command to check SoO extended community filter information.
- Run the **display bgp routing-table as-path-filter *as-path-filter-number*** command to check information about routes matching a specified AS_Path filter.

- Run the **display bgp routing-table community-filter { community-filter-name | basic-community-filter-number } [whole-match]** command to check information about routes matching a specified BGP community filter.
- Run the **display bgp routing-table peer ipv4-address advertised-routes [statistics]** command to check information about routes advertised by a BGP device to its peers.
- Run the **display bgp routing-table peer ipv4-address received-routes [active] [statistics]** command to check information about routes received by a BGP device from its peers.
- Run the **display bgp routing-table peer ipv4-address received-routes network { mask | mask-length } original-attributes** command to check information about the original attributes of the routes received from the specified peer.

----End

1.1.9.2.7 Configuring BGP XPL

Using XPL to Filter the BGP Routes to Be Advertised

A BGP device can use a route-filter to filter the routes to be advertised and modify route attributes to control the network traffic forwarding path.

Usage Scenario

BGP is used to transmit routing information between ASs, and route advertisement directly affects traffic forwarding.

In most cases, a BGP routing table has a large number of routes. Transmitting them will intensify the load of a device. To address this problem, configure the device to advertise only desired routes.

In addition, multiple routes to the same destination may exist and travel through different ASs. To direct routes to specific ASs, you also need to filter the routes to be advertised.

You can use XPL route-filters to control the BGP routes to be advertised to all peers or peer groups or to a specified peer or peer group.

Pre-configuration Tasks

Before using XPL to filter the BGP routes to be advertised, [configure basic BGP functions](#).

Procedure

- Use XPL to filter the BGP routes to be advertised to all peers or peer groups.
Perform the following steps on the BGP device:
 - Run **system-view**
The system view is displayed.
 - Run **bgp as-number**

- The BGP view is displayed.
- c. Run **ipv4-family unicast**
- The IPv4 unicast address family view is displayed.
- d. Run **route-filter route-filter-name export [direct | isis process-id | ospf process-id | rip process-id | static]**
- The BGP device is configured to filter the routes to be advertised to all peers or peer groups.
- e. Run **commit**
- The configuration is committed.
- Use XPL to filter the BGP routes to be advertised to a specific peer or peer group.
- Perform the following steps on the BGP device:
- a. Run **system-view**
- The system view is displayed.
- b. Run **bgp as-number**
- The BGP view is displayed.
- c. Run **ipv4-family unicast**
- The IPv4 unicast address family view is displayed.
- d. Run **peer { group-name | ipv4-address } route-filter route-filter-name export**
- The BGP device is configured to filter the routes to be advertised to the specified peer or peer group.
- e. Run **commit**
- The configuration is committed.

----End

Verifying the Configuration

Run the following commands to check configurations:

- Run the **display xpl route-filter state [attached | unattached]** command to check information about the configured route-filter.
- Run the **display bgp routing-table [peer ipv4-address advertised-routes [statistics]]** command to check the routes in the BGP routing table.

Using XPL to Filter the BGP Routes to Be Received

A BGP device can use a route-filter to filter the routes to be received and modify route attributes to control the network traffic forwarding path.

Usage Scenario

BGP is used to transmit routing information between ASs, and route advertisement directly affects traffic forwarding.

A BGP device may receive multiple routes to the same destination network from different peers. To control the network traffic forwarding path, filter the routes to be received.

In addition, a BGP device may receive a large number of routes during an attack, which may consume lots of device resources. In this case, the administrator must limit the resource consumption based on the network planning and device performance.

You can use XPL route-filters to control the BGP routes to be received from all peers or peer groups or from a specified peer or peer group.

Pre-configuration Tasks

Before using XPL to filter the BGP routes to be received, [configure basic BGP functions](#).

Procedure

- Use XPL to filter the BGP routes to be received from all peers or peer groups.

Perform the following steps on the BGP device:

 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - d. Run **route-filter route-filter-name import**
The BGP device is configured to filter the routes to be received from all peers or peer groups.
 - e. Run **commit**
The configuration is committed.
- Use XPL to filter the BGP routes to be received from a specific peer or peer group.

Perform the following steps on the BGP device:

 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - d. Run **peer { group-name | ipv4-address } route-filter route-filter-name import**

The BGP device is configured to filter the routes to be received from the specified peer or peer group.

e. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the following commands to check configurations:

- Run the **display xpl route-filter state [attached | unattached]** command to check information about the configured route-filter.
- Run the **display bgp routing-table [peer *ipv4-address* received-routes [statistics]]** command to check the routes in the BGP routing table.

1.1.9.2.8 Configuring BGP Route Summarization

Configuring BGP route summarization on a device can reduce the sizes of routing tables on the peers of the device.

Usage Scenario

The BGP routing table of the router on a medium or large BGP network contains a large number of routing entries. Storing the routing table consumes a large number of memory resources, and transmitting and processing routing information consume lots of network resources. Configuring route aggregation can reduce the size of a routing table, prevent specific routes from being advertised, and minimize the impact of route flapping on network performance. BGP route summarization and routing policies enable BGP to effectively transmit and control routes.

BGP supports automatic and manual summarization. Manual summarization takes precedence over automatic summarization.

Pre-configuration Tasks

Before configuring BGP route summarization, complete the following task:

- [Configure basic BGP functions.](#)

Procedure

- Configure automatic route summarization.

a. Run **system-view**

The system view is displayed.

b. Run **bgp *as-number***

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **summary automatic**

Automatic summarization is configured for imported routes.

The **summary automatic** command summarizes routes imported by BGP. The routes can be direct routes, static routes, RIP routes, OSPF routes, or IS-IS routes. After this command is run, BGP summarizes routes based on natural network segments. The command, however, cannot summarize routes imported using the **network** command.

e. Run **commit**

The configuration is committed.

- Configure manual route summarization.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

- Configure manual summarization according to the actual networking.

- To advertise both the summary route and specific routes, run the **aggregate ipv4-address { mask | mask-length }** command.
- To advertise only the summary route, run the **aggregate ipv4-address { mask | mask-length } detail-suppressed** command.
- To advertise a specific route, run the **aggregate ipv4-address { mask | mask-length } suppress-policy route-policy-name** command.
To advertise a specific route, you can also run the **peer route-policy** command.
- To generate a summary route used to detect a loop, run the **aggregate ipv4-address { mask | mask-length } as-set** command.
- To set the attributes of a summary route, run the **aggregate ipv4-address { mask | mask-length } attribute-policy route-policy-name** command.
To set the attributes of a summary route, you can also run the **peer route-policy** command.
If **as-set** is set in the **aggregate** command and the **apply as-path** command is run to set the AS_Path attribute, the AS_Path attribute does not take effect.
- To generate a summary route according to certain specific routes, run the **aggregate ipv4-address { mask | mask-length } origin-policy route-policy-name** command.

Manual summarization is valid for the existing routing entries in the local BGP routing table. For example, if a route with the mask length longer than 16, such as 10.1.1.0/24, does not exist in the BGP routing table, BGP

does not advertise the summary route even after the **aggregate 10.1.1.1 16** command is run.

e. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display bgp routing-table [network [mask | mask-length]]** command to check information about BGP summary routes.

1.1.9.2.9 Configuring BGP to Generate a Summary Default Route

You can configure BGP to generate a summary default route and then determine whether to advertise the default route to a peer by using a route policy.

Usage Scenario

On the network shown in [Figure 1-384](#), a BGP-VPNv4 peer relationship is established between PE1 and PE2. In addition, an EBGP-VPN peer relationship is established between PE2 and the core network device, and another EBGP-VPN peer relationship is established between PE1 and CE1. The core network device advertises routes to PE2 through the EBGP-VPN peer relationship. PE2 then advertises the received routes to PE1 through the BGP-VPNv4 peer relationship. Upon receipt, PE1 leaks the routes to its VPN instance. A summary default route can be generated only if routes with a specific prefix exist among the leaked routes. You can configure BGP to generate a summary default route on PE1 so that PE1 matches the leaked routes against an IP prefix list and summarizes the matched routes into a default route. As a result, PE1 advertises only the summary default route to CE1. Traffic over the unmatched routes will not be diverted to PE1, thereby conserving PE1's bandwidth resources.

Figure 1-384 Typical networking



Pre-configuration Tasks

Before configuring BGP to generate a summary default route, [configure basic BGP functions](#).

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **ip ip-prefix ip-prefix-name [index index-number] matchMode ipv4-address masklen [match-network] [greater-equal greater-equal-value] [less-equal less-equal-value]**

An IPv4 prefix list is configured.

Step 3 Run **bgp as-number**

The BGP view is displayed.

Step 4 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 5 Run **aggregate default-route origin-ip-prefix ip-prefix-name [attribute-policy attribute-policy-name]**

BGP is enabled to generate a summary default route based on an IPv4 prefix list.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display bgp routing-table [network [mask | mask-length]]** command to check information about BGP summary default routes.

1.1.9.2.10 Configuring a BGP Peer Group

Configuring BGP peer groups simplifies the BGP network configuration and improves the route advertisement efficiency.

Usage Scenario

A BGP peer group consists of BGP peers that have the same update policies and configurations.

A large-scale BGP network has a large number of peers. Configuring and maintaining these peers is difficult. To address this problem, configure a BGP peer group for BGP peers with the same configurations. Configuring BGP peer groups simplifies peer management and improves the route advertisement efficiency.

Based on the ASs where peers reside, peer groups are classified as follows:

- IBGP peer group: The peers of an IBGP peer group are in the same AS.
- Pure EBGP peer group: The peers of a pure EBGP peer group are in the same external AS.
- Mixed EBGP peer group: The peers of a mixed EBGP peer group are in different external ASs.

If a function is configured on a peer and its peer group, the function configured on the peer takes effect. After a peer group is created, peers can be added to the peer group. If these peers are not configured separately, they will inherit the configurations of the peer group. If a peer in a peer group has a specific configuration requirement, the peer can be configured separately. The configuration of this peer will override the configuration that the peer has inherited from the peer group.

Pre-configuration Tasks

Before configuring BGP peer groups, [configure basic BGP functions](#).

Creating an IBGP Peer Group

If multiple IBGP peers exist, adding them to an IBGP peer group can simplify the BGP network configuration and management. When creating an IBGP peer group, you do not need to specify an AS number for the IBGP peer group.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **group group-name internal**

An IBGP peer group is created.

Step 4 Run **peer ipv4-address group group-name**

A peer is added to the peer group.



NOTE

You can repeat step 4 to add multiple peers to the peer group. If the local device has not established a peer relationship with this peer, the device will attempt to establish a peer relationship with this peer, and set the AS number of this peer to the AS number of the peer group.

When creating an IBGP peer group, you do not need to specify the AS number.

After configuring a peer group, you can configure BGP functions for the peer group. By default, all peers in a peer group inherit the entire configuration of the peer group. If you configure a peer in a peer group independently. The configuration of this peer will override the configuration that the peer has inherited from the peer group.

Step 5 (Optional) Run **peer group-name description description-text**

A description is configured for the peer group.

You can simplify network management by configuring the descriptions.

Step 6 Run **commit**

The configuration is committed.

----End

Creating a Pure EBGP Peer Group

If multiple EBGP peers exist in an AS, adding them to an EBGP peer group can simplify the BGP network configuration and management. All the peers in a pure EBGP peer group must have the same AS number.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **group group-name external**

An EBGP peer group is created.

Step 4 Run **peer group-name as-number as-number**

An AS number is set for the peer group.

If peers already exist in a peer group, the AS number of the peer group cannot be changed.

Step 5 Run **peer ipv4-address group group-name**

A peer is added to a peer group.



You can repeat Step 5 to add multiple peers to the peer group. If the specified peer does not exist, the system automatically creates it in the BGP view and sets the AS number of the peer to that of the peer group.

After a peer group is created, you can configure BGP functions for the peer group in batches. By default, each peer in a peer group inherits the configurations of the peer group. However, if you configure a member peer separately, the configuration of this peer will override that inherited from the peer group.

Step 6 (Optional) Run **peer group-name description description-text**

A description is configured for the peer group.

You can simplify network management by configuring a description.

Step 7 Run **commit**

The configuration is committed.

----End

Creating a Mixed EBGP Peer Group

If multiple EBGP peers exist in different ASs, adding them to a mixed EBGP peer group can simplify the BGP network configuration and management. When

creating a mixed EBGP peer group, you need to specify an AS number for each peer.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **group group-name external**

A mixed EBGP peer group is created.

Step 4 Run **peer ipv4-address as-number as-number**

A peer is created, and an AS number is set for this peer.

Step 5 Run **peer ipv4-address group group-name**

A peer is added to the peer group.



You can repeat Steps 4 and 5 to add multiple peers to the peer group.

You need to specify an AS number for each peer in a mixed EBGP peer group.

After configuring a peer group, you can configure BGP functions for the peer group. By default, all peers in a peer group inherit the entire configuration of the peer group. If you configure a peer in a peer group independently. The configuration of this peer will override the configuration that the peer has inherited from the peer group.

Step 6 (Optional) Run **peer group-name description description-text**

A description is configured for the peer group.

You can simplify network management by configuring the descriptions.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the BGP Peer Group Configuration

After configuring a BGP peer group, check information about BGP peers and BGP peer groups.

Prerequisites

A BGP peer group has been configured.

Procedure

- Run the **display bgp peer [*ipv4-address*] verbose** command to check detailed information about BGP peers.
- Run the **display bgp group [*group-name*]** command to check information about BGP peer groups.

NOTE

This command is applied only to devices on which BGP peer groups are created.

If a peer group is specified in this command, detailed information about this peer group will be displayed. If no peer group is specified in this command, information about all BGP peer groups is displayed.

----End

1.1.9.2.11 Configuring Distributed BGP Peers

In scenarios with a large number of peers and routes, devices must have high peer and route management capabilities. To ensure stable peer relationships and fast route selection in such scenarios, distributed peers need to be configured.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp *as-number***

The BGP view is displayed.

Step 3 Run **distribute-instance *distribute-instance-name* [**os-group *os-group-name***]**

A distributed BGP instance is created.

Step 4 Run **peer { *ip/peer/pv4Addr|peer/pv6Addr* } **as-number** **peer-as** **distribute-instance *distribute-name*****

A peer is added to the specified distributed instance.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, you can run the **display bgp peer** command to view information about BGP peers.

1.1.9.2.12 Configuring BGP Parallelization

BGP parallelization supports multi-thread concurrent route parsing, which accelerates route parsing, speeds up route matching against local import policies, and improves the performance of BGP route learning through complex policies. When there are a large number of BGP routes and route attributes and route

filtering policies are complex, BGP route learning in single-thread mode is slow. In this case, you can configure BGP parallelization to improve performance.

Context

The core idea of BGP parallelization is to allocate computing tasks to as many processors as possible, use the parallel computing capability of multiple processors, and divide the routing table and routing information into multiple subtasks for processing. Different subtasks can be allocated to different processors for parallel computing, improving computing efficiency. Then, multiple processors communicate and exchange data to accelerate routing information processing. In this way, the multi-thread mechanism is used to improve performance during BGP parallelization.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp route-learning acceleration enable**

BGP parallelization is enabled so that BGP processes learn routes in parallel.

Step 3 (Optional) Run **bgp route-learning thread-number number**

The number of BGP threads is specified. A larger number of BGP threads does not mean higher BGP route learning performance. It depends on the number of CPU cores. You are advised to use the default number of threads.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.9.2.13 Configuring a BGP Route Reflector

By configuring a BGP route reflector (RR), you can avoid fully meshed connections between multiple IBGP peers.

Usage Scenario

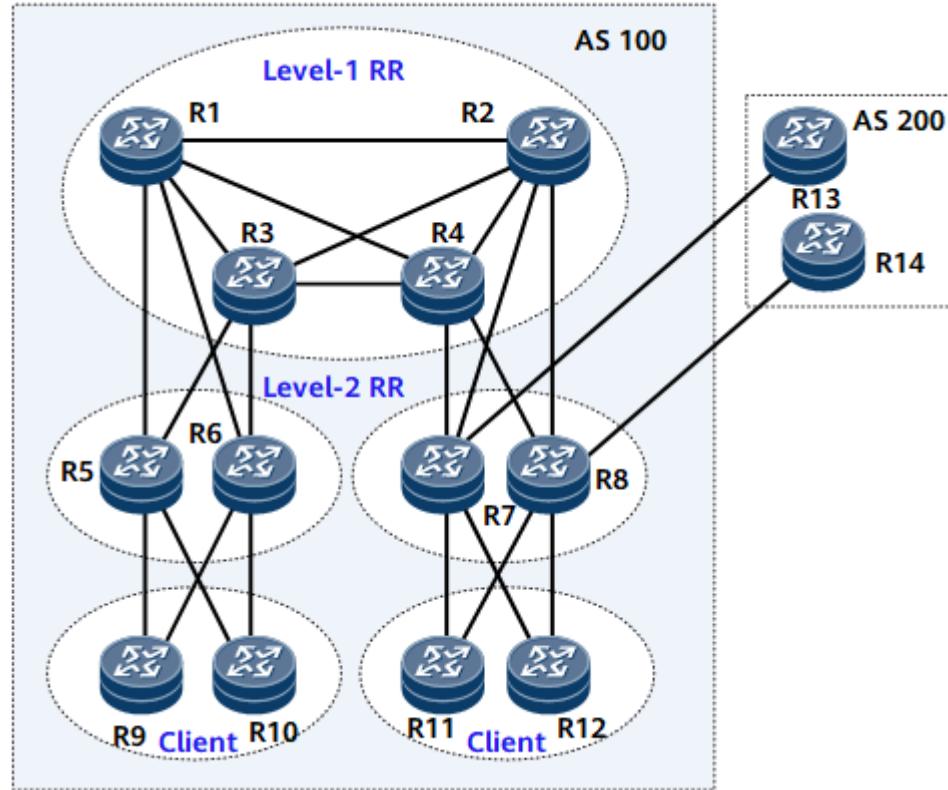
Fully meshed connections need to be established between IBGP peers to ensure the connectivity between IBGP peers. If there are n routers in an AS, $n \times (n-1)/2$ IBGP connections need to be established. When there are a lot of IBGP peers, network resources and CPU resources are greatly consumed. Route reflection can solve the problem.

RRs can reduce the total number of IBGP connections. On a large network, to reduce the number of clients of each route reflector, you need to configure multiple RRs. Because fully meshed connections need to be established between RRs, there are still a large number of IBGP connections on the network. In this situation, configure hierarchical RRs to further reduce the number of IBGP connections.

[Figure 1-385](#) shows typical networking with hierarchical RRs. In this networking, R1, R2, R3, and R4 function as Level-1 RRs; R5, R6, R7, and R8 function as level-2

RRs and the clients of level-1 RRs. Level-1 RRs are not the clients of any RR and must be fully meshed. Level-2 RRs function as the clients of Level-1 RRs and do not need to be fully meshed.

Figure 1-385 Typical networking with hierarchical RRs



Pre-configuration Tasks

Before configuring a BGP route reflector, [configure basic BGP functions](#).

Configuring a Route Reflector and Specifying Clients

RRs reflect routes between clients, and therefore IBGP connections do not need to be established between the clients.

Context

In an AS, one router functions as an RR, the other routers function as clients. IBGP connections are established between the RR and its clients. The RR and its clients form a cluster. The RR transmits (or reflects) routes between clients, and clients do not need to establish IBGP connections.

An RR simplifies configurations because all configurations need to be configured only on the RR and clients do not need to know that they are clients.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

Step 4 Run **peer { group-name | ipv4-address } reflect-client**

The device is configured as an RR, and a client is specified for it.

The router on which the **peer reflect-client** command is run functions as the RR, and the specified peer or peer group functions as a client.

 **NOTE**

The configurations of RRs and clients in an address family are valid only in this address family and cannot be inherited by other address families. Therefore, configure RRs and clients in the required address family.

Step 5 Run **commit**

The configuration is committed.

----End

(Optional) Disabling Route Reflection Between Clients Through the RR

If the clients of a route reflector are fully connected, you need to disable route reflection among them through the RR to reduce bandwidth consumption.

Context

On some networks, if fully meshed IBGP connections have been established between clients, they can directly exchange routing information. Therefore, route reflection between clients through the RR is unnecessary, which also occupies bandwidth. In this case, disable route reflection among them through the RR.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **undo reflect between-clients**

Route reflection is disabled among clients through the RR.

If the clients of the route reflector are fully connected, you can use the **undo reflect between-clients** command to disable route reflection among the clients through the RR. This command is applicable to only the route reflector.

Step 5 Run commit

The configuration is committed.

----End

(Optional) Setting the Cluster ID of a Route Reflector

When there are multiple route reflectors in a cluster, you need to configure the same cluster ID for all the route reflectors in this cluster to avoid routing loops.

Context

To enhance network reliability and avoid single points of failure, more than one route reflector can be configured in a cluster. In this case, you need to set the same cluster ID for all the route reflectors in the same cluster. This can reduce the number of routes to be received by each route reflector and save the memory.

NOTE

To ensure that a client can learn the routes reflected by a route reflector, the cluster ID of the route reflector must be different from the router ID of the client. If they are the same, the client discards the received routes.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run bgp *as-number*

The BGP view is displayed.

Step 3 Run ipv4-family unicast

The IPv4 unicast address family view is displayed.

Step 4 Run reflector cluster-id { *cluster-id-value* | *cluster-id-ipv4* }

The cluster ID of a route reflector is set.

When there are multiple route reflectors in a cluster, you need to run the command to configure the same *cluster-id* for all the route reflectors in this cluster.

The **reflector cluster-id** command can be configured on only route reflectors.

Step 5 Run commit

The configuration is committed.

----End

(Optional) Preventing a Device from Adding BGP Routes to the IP Routing Table

After you prevent an RR from adding BGP routes to the IP routing table, the RR no longer forwards traffic, which improves route advertisement efficiency.

Context

In most cases, BGP routes are added to the IP routing table of the router for traffic forwarding. If the router does not need to forward traffic, prevent it from adding BGP routes to the IP routing table.

RRs need to be prevented from adding BGP routes to the IP routing table. An RR not only transmits routes but also forwards traffic. If the RR is connected to many clients and non-clients, the route transmission task will consume a lot of CPU resources of the RR, and as a result, the RR cannot forward traffic. To improve the route transmission efficiency, prevent the RR from adding BGP routes to the IP routing table so that the RR only transmits routes.

Perform the following steps on the router that runs BGP:

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **routing-table rib-only [route-policy route-policy-name | route-filter route-filter-name]**

The device is prevented from adding BGP routes to the IP routing table.

If **route-policy route-policy-name** or **route-filter route-filter-name** is configured in the **routing-table rib-only** command, routes matching the policy are not added to the IP routing table, and routes that do not match the policy are added to the IP routing table, with the route attributes unchanged.



The **routing-table rib-only** and **active-route-advertise** commands are mutually exclusive.

Step 5 Run **commit**

The configuration is committed.

----End

(Optional) Enabling an RR to Modify Route Attributes Using an Export Policy

You can enable an RR to modify route attributes using an export policy to control BGP route selection.

Context

Route attributes cannot be modified using an export policy on an RR because it would cause routing loops. By default, an RR is disabled from modifying route attributes based on an export policy. However, if you need to re-plan the network

traffic, you can enable the RR to modify route attributes based on an export policy.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **reflect change-path-attribute**

The RR is configured to modify path attributes of BGP routes based on an export policy.

After the **reflect change-path-attribute** command is run on an RR, the configuration of modifying path attributes of routes based on an export policy takes effect. The configurations are as follows:

- **apply as-path**: modifies the AS_Path attribute of BGP routes.
- **apply comm-filter delete**: deletes community attributes from BGP routes.
- **apply community**: modifies the community attribute of BGP routes.
- **apply large-community**: modifies the Large-community attribute of BGP routes.
- **apply cost**: modifies the MED value of BGP routes.
- **apply ip-address nexthop**: modifies the next-hop IP address of BGP routes.
- **apply local-preference**: modifies the Local_Pref value of BGP routes.
- **apply origin**: modifies the Origin attribute of BGP routes.
- **apply extcommunity**: modifies the VPN target extended community attribute of BGP routes.
- **apply extcommunity soo { site-of-origin } &<1-16> additive**: modifies the SoO extended community attribute of BGP routes.

NOTE

After the **reflect change-path-attribute** command is run on the RR, the **peer route-policy export** command takes precedence over the **peer next-hop-invariable** and **peer next-hop-local** commands.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the BGP Route Reflector Configuration

After configuring a BGP route reflector, verify information about BGP routes and BGP peer groups.

Prerequisites

A BGP route reflector has been configured.

Procedure

- Run the **display bgp group [group-name]** command to check information about BGP peer groups.
- Run the **display bgp routing-table [network] [mask | mask-length] [longer-prefixes]** command to check information about the BGP routing table.

----End

1.1.9.2.14 Configuring a BGP Confederation

On a large BGP network, configuring a BGP confederation reduces the number of IBGP connections and simplifies routing policy management, which increases the route advertisement efficiency.

Usage Scenario

A confederation is a solution to the increasing number of IBGP connections in an AS. The confederation divides an AS into multiple sub-ASs. In each sub-AS, IBGP peer relationships are set up or an RR is configured on one of the IBGP peers. EBGP connections are set up between sub-ASs.



Compared with RRs, confederations facilitate IGP extensions.

Pre-configuration Tasks

Before configuring a BGP confederation, complete the following tasks:

- Configure link layer protocol parameters and assigning IP addresses to the interfaces to ensure that the status of the link layer protocol of the interface is Up.
- [Configure basic BGP functions](#).

Procedure

- Configure a BGP Confederation.

Perform the following steps on the BGP device:

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **confederation id as-number**

A confederation ID is set.

d. Run **confederation peer-as { as-number } &<1-32>**

The number of the sub-AS where other EBGP peers connected to the local AS reside is specified.

The *as-number* used to specify a sub-AS in a confederation is valid in the confederation.

The **confederation id** and **confederation peer-as** commands must be run for all EBGP peers in a confederation, and the EBGP peers must have the same confederation ID.

 NOTE

The old speaker supporting 2-byte AS numbers and the new speaker supporting 4-byte AS numbers cannot exist in the same confederation. Otherwise, routing loops may occur because AS4_Path does not support confederations.

e. Run **commit**

The configuration is committed.

- Configure the compatibility of the confederation.

If some routers in a confederation do not comply with the standard protocols, you can perform the following steps so that the local device is compatible with them:

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **confederation nonstandard**

Confederation compatibility is configured.

d. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp peer [ipv4-address] verbose** command to check detailed peer information.
- Run the **display bgp routing-table [network] [mask | mask-length] [longer-prefixes]** command to check information about routes in the BGP routing table.

1.1.9.2.15 Configuring BGP Community Attributes

Community attributes simplify routing policy management.

Usage Scenario

Community attributes are used to simplify routing policy application and facilitate network maintenance. They allow a group of BGP devices in different ASs to share the same routing policies. Before advertising a route with the community attribute to peers, a BGP device can change the original community attribute of this route. Community attributes are route attributes, which are transmitted between BGP peers, and the transmission is not restricted within an AS.

Pre-configuration Tasks

Before configuring BGP community attributes, [configure basic BGP functions](#).

Configuring a Community Attribute-Related Policy

A policy that references a community attribute needs to be configured before the community attribute is set for routing information.

Procedure

- Configure a community attribute-based route-policy.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **route-policy route-policy-name matchMode node node**
A route-policy with a node is created, and the route-policy view is displayed.
 - c. (Optional) Configure if-match clauses for the route-policy. Community attributes can be added only to the routes that match if-match clauses, and the community attributes of only the routes that match the if-match clauses can be modified. For details about the configuration, see [\(Optional\) Configuring an if-match Clause](#).
 - d. Configure community or extended community attributes for BGP routes. This section describes only common configurations. For details, see [\(Optional\) Configuring an apply Clause](#).
 - To configure community attributes for BGP routes, run the **apply community { cmntyValue | cmntyNum | internet | no-advertise | no-export | no-export-subconfed } &<1-32> [additive]** command.

NOTE

A maximum of 32 community attributes can be configured using this command at a time.

- To configure the BGP VPN-Target extended community attribute, run the **apply extcommunity { rt extCmntyValue } &<1-16> [additive]** command.
- To configure the SoO extended community attribute for BGP routes, run the **apply extcommunity soo { site-of-origin } &<1-16> additive** command.

- e. Run **commit**

The configuration is committed.

- Configure a community attribute-based route-filter.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **xpl route-filter** *route-filter-name* or **edit xpl route-filter** *route-filter-name*

A route-filter is created, and the route-filter view is displayed.
 - c. (Optional) Configure XPL common and condition clauses. Community attributes can be added only to the routes that match XPL clauses, and the community attributes of only the routes that match the XPL clauses can be modified. For details about the configuration, see [Common Clauses](#) and [Condition Clauses](#) in "XPL Configuration."
 - d. Configure community or extended community attributes for BGP routes.
 - Configure community attributes for BGP routes. For configuration details, see [Action Clauses Used to Set Community Attributes for BGP Routes](#) in "XPL Configuration."
 - Configure the Link Bandwidth extended community attribute for BGP routes. For configuration details, see [Setting a Link Bandwidth Extended Community Attribute for BGP Routes](#) in "XPL Configuration."
 - e. Run **commit**

The configuration is committed.

----End

Configuring Community Attribute Advertisement

A community attribute defined in a routing policy takes effect only after community attribute advertisement is configured.

Procedure

- Use a route-policy to define specific community attributes when configuring BGP community attribute advertisement.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bgp** *as-number*

The BGP view is displayed.
 - c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.
 - d. Run **peer { ipv4-address | group-name } route-policy** *route-policy-name export*

An export route-policy is configured.
 - e. Run **peer { ipv4-address | group-name } advertise-community**

The device is configured to advertise standard community attributes to the specified peer.

By default, a device does not advertise community attributes to any peer.

f. Run **commit**

The configuration is committed.

- Use a route-filter to define specific community attributes when configuring BGP community attribute advertisement.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } route-filter route-filter-name export**

An export route-filter is configured.

e. Run **peer { ipv4-address | group-name } advertise-community**

The device is configured to advertise standard community attributes to the specified peer.

f. Run **commit**

The configuration is committed.

----End

Configuring the Device to Advertise Extended Community Attributes

The extended community attributes defined in a route-policy take effect only after the extended community attributes are advertised.

Procedure

- Use a route-policy to define specific extended community attributes before configuring BGP to advertise the extended community attributes.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } route-policy route-policy-name export**

An export route-policy is configured.

e. Run **peer { ipv4-address | group-name } advertise-ext-community**

The device is configured to advertise extended community attributes to the specified peer.

If a peer or peer group only needs to accept routes but not extended community attributes, you can run the **peer discard-ext-community** command for the peer or peer group to discard the extended community attributes carried in received routes. If the peer or peer group only needs to discard the RPKI BGP origin AS validation result, specify **origin-as-validation** in the command.

f. (Optional) Run **peer { ipv4-address | group-name } advertise ebgp link-bandwidth**

The device is configured to advertise the link bandwidth extended community attribute to the specified EBGP peer.

g. (Optional) Run **peer { ipv4-address } advertise link-bandwidth transitive**

The device is configured to convert the link bandwidth extended community attribute (optional non-transitive) carried in BGP routes into an optional transitive attribute before advertising the routes to the specified peer. If the device changes the next-hop address of a received route carrying the link bandwidth extended community attribute to its own address, the device deletes this attribute before advertising the route to the specified peer.

h. Run **commit**

The configuration is committed.

- Use a route-filter to define specific extended community attributes before configuring BGP to advertise the extended community attributes.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

d. Run **peer { ipv4-address | group-name } route-filter route-filter-name export**

An export route-filter is configured.

e. Run **peer { ipv4-address | group-name } advertise-ext-community**

The device is configured to advertise extended community attributes to the specified peer.

After the **peer advertise-ext-community** command is run, BGP advertises the routes with extended community attributes to the specified peer. If the peer only needs to accept the routes but not the extended community attributes, you can run the **peer discard-ext-community** command on the peer to discard the extended community attributes in the received routing information.

- f. (Optional) Run **peer { ipv4-address | group-name } advertise ebgp link-bandwidth**

The device is configured to advertise the link bandwidth extended community attribute to the specified EBGP peer.

- g. (Optional) Run **peer ipv4-address } advertise link-bandwidth transitive**

The device is configured to convert the link bandwidth extended community attribute (optional non-transitive) carried in BGP routes into an optional transitive attribute before advertising the routes to the specified peer. If the device changes the next-hop address of a received route carrying the link bandwidth extended community attribute to its own address, the device deletes this attribute before advertising the route to the specified peer.

- h. Run **commit**

The configuration is committed.

----End

Verifying the BGP Community Attribute Configuration

After configuring BGP community attributes, verify the configured BGP community attributes.

Prerequisites

A BGP community attribute has been configured.

Procedure

- Run the **display bgp routing-table network [mask | mask-length]** command to check information about BGP routes.
- Run the **display bgp routing-table community [communityNum | strCommunityNum | internet | no-advertise | no-export | no-export-subconfed | g-shut] &<1-33> [whole-match]** command to check information about routes with a specified BGP community attribute.

----End

1.1.9.2.16 Configuring the BGP Large-Community Attribute

The Large-Community attribute can be used to flexibly apply route-policies.

Usage Scenario

The Large-Community attribute can represent a 2-byte or 4-byte Autonomous System Number (ASN), and has two 4-byte LocalData IDs. The administrator can therefore apply route-policies more flexibly. The Large-Community attribute extends and can be used together with a community attribute.

Pre-configuration Tasks

Before configuring the BGP Large-Community attribute, [configure basic BGP functions](#).

Configuring a Route-Policy Related to the Large-Community Attribute

Before configuring the Large-Community attribute for routes, configure a route-policy in which the Large-Community attribute is applied.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **route-policy route-policy-name matchMode node node**

A route-policy with a node is created, and the route-policy view is displayed.

Step 3 (Optional) Configure filtering conditions (if-match clauses) for the route-policy. Large-Community values can be added only to the BGP routes that pass the filtering, and the Large-Community values of only these routes can be modified.

For details, see [\(Optional\) Configuring an if-match Clause](#).

Step 4 Run **apply large-community { aa:bb:cc } &<1-16> { additive | overwrite | delete }** or **apply large-community-list large-community-list-name { additive | overwrite | delete }**

Large-Community values are configured for BGP routes.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring the Device to Advertise the Large-Community Attribute

The large-community attribute defined in a routing policy takes effect only after the large-community attribute is advertised.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **peer { ipv4-address | group-name } route-policy route-policy-name export**

An export route-policy is configured.

 NOTE

When configuring the BGP Large-Community attribute, use a route-policy to define specific Large-Community values, and then apply the route-policy to the routes to be advertised.

For details about the route-policy configuration, see the chapter "Routing Policy Configuration".

Step 5 Run `peer { ipv4-address | group-name } advertise-large-community`

The device is enabled to advertise the Large-Community attribute to a peer or peer group.

Step 6 Run `commit`

The configuration is committed.

----End

Verifying the Large-Community Configuration

After configuring the BGP Large-Community attribute, verify the configuration.

Prerequisites

All configurations of the BGP Large-Community attribute have been performed.

Procedure

- Run the **display bgp routing-table network [mask | mask-length]** command to check information about BGP routes.
- Run the **display bgp routing-table large-community [aa:bb:cc] &<1-33> [whole-match]** command to check information about the routes with the specified BGP Large-Community attribute.

----End

1.1.9.2.17 Configuring Prefix-based BGP ORF

After prefix-based BGP outbound route filtering (ORF) is configured, the local device sends its prefix-based import policy to a peer so that the peer filters routes during route advertisement.

Usage Scenario

When a device expects to receive only required routes from the remote device and the remote end does not want to maintain a separate export policy for each connected peer, you can configure prefix-based ORF which supports on-demand route advertisement.

Pre-configuration Tasks

Before configuring prefix-based BGP ORF, complete the following tasks:

- [Configure basic BGP functions.](#)
- [Configure an IPv4 prefix list.](#)

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 (Optional) Run **orrf-limit limit-value**

The maximum number of ORFs that can be accepted from a peer is set.

Step 4 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 5 Run **peer { group-name | ipv4-address } ip-prefix ip-prefix-name import**

An IP prefix list-based import policy is configured to filter routes received from the specified peer or peer group.

Step 6 Run **peer { group-name | ipv4-address } capability-advertise orf [non-standard-compatible] ip-prefix { both | receive | send }**

Prefix-based ORF is configured for a BGP peer or peer group.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp peer ipv4-address verbose** command to check detailed information about BGP peers.
- Run the **display bgp peer ipv4-address orf ip-prefix** command to check prefix-based ORF information received from a specified peer.

1.1.9.2.18 Adjusting the BGP Network Convergence Speed

You can adjust the BGP network convergence speed by adjusting BGP peer connection parameters to adapt to changes on large-scale networks.

Usage Scenario

BGP is used to transmit routing information on large-scale networks. Frequent network changes affect the establishment and maintenance of BGP peer relationships, which in turn affects the BGP network convergence speed.

The route dampening and triggered update functions of BGP suppress frequent route changes to a certain extent, but cannot minimize the impact of network flapping on BGP connections.

You can configure BGP timers and disable rapid EBGP connection reset to suppress BGP network flapping and speed up BGP network convergence.

- BGP Keepalive and Hold timers
BGP uses Keepalive messages to maintain BGP peer relationships and monitor connection status. After establishing a BGP connection, two peers send Keepalive messages periodically to each other to detect the BGP connection status based on the Keepalive timer. If a router does not receive any Keepalive message or any other types of packets from its peer within the hold time set by the Hold timer, the router considers the BGP connection interrupted and terminates the BGP connection.
- BGP MinRouteAdvertisementIntervalTimer
BGP does not periodically update a routing table. When BGP routes change, BGP updates the changed BGP routes in the BGP routing table by sending Update messages. If a route changes frequently, to prevent the router from sending Update messages upon every change, set the interval at which Update messages are sent.
- Rapid EBGP peer reset
Rapid EBGP connection reset is enabled by default so that EBGP can quickly detect the status of interfaces used to establish EBGP connections. If the interface status changes frequently, you can disable rapid EBGP connection reset so that direct EBGP sessions will not be reestablished and deleted as interface alternates between Up and Down, which speeds up network convergence.
- BGP ConnectRetry Timer
After BGP initiates a TCP connection, the ConnectRetry timer will be stopped if the TCP connection is established successfully. If the first attempt to establish a TCP connection fails, BGP attempts to establish the TCP connection again after the ConnectRetry timer expires.
Setting a short ConnectRetry interval reduces the period BGP waits between attempts to establish a TCP connection, which speeds up the establishment of the TCP connection.
Setting a long ConnectRetry interval suppresses routing flapping caused by frequent peer relationship flapping.

After slow peer detection is configured on a device, the device identifies the slow BGP peer (if any) and removes it from the update peer-group to prevent this slow peer from affecting route advertisement to other peers in this update peer-group. Slow peer detection speeds up BGP network convergence.

Pre-configuration Tasks

Before setting parameters for a BGP peer connection, [configure basic BGP functions](#).

Configuring BGP Keepalive and Hold Timers

The values of BGP Keepalive and Hold timers determine the speed at which BGP detects network faults. You can adjust the values of these timers to improve network performance.

Context

BGP uses Keepalive messages to maintain peer relationships. After establishing a BGP connection, two peers periodically send Keepalive messages to each other to

detect BGP peer relationship status. If a device receives no Keepalive message from its peer after the Hold timer expires, the device considers the BGP connection interrupted.

- If short Keepalive time and holdtime are set, BGP can fast detect link faults. This speeds up BGP network convergence, but increases the number of Keepalive messages on the network and loads of routers, and consumes more network bandwidth resources.
- If long Keepalive time and holdtime are set, the number of Keepalive messages on the network and loads of routers are reduced. If the Keepalive time is too long, BGP cannot fast detect link status changes, which slows down BGP network convergence and may cause packet loss.

NOTICE

Changing timer values using the **timer** command or the **peer timer** command interrupts BGP peer relationships between routers. Therefore, exercise caution before running either of the command.

Keepalive and Hold timers can be configured either for all peers or peer groups, or for a specific peer or peer group. Keepalive and Hold timers configured for a specific peer take precedence over those configured for the peer group of this peer. In addition, Keepalive and Hold timers configured for a specific peer or peer group take precedence over those configured for all peers or peer groups.

Procedure

- Configure BGP timers for all peers or peer groups.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run the **timer keepalive keepalive-time hold hold-time [min-holdtime min-hold-value]** command to set the BGP keepalive time and hold time, or run the **timer send-hold send-hold-time** command to set the hold time during which the local device does not proactively disconnect from the peer device.

The proper maximum interval at which Keepalive messages are sent is one third the hold timer and cannot be less than 1s. If the hold time is not set to 0, it is 3s at least.

NOTE

You are advised to configure *hold-time* according to the total number of peers in all BGP address families. The more peers, the larger the recommended minimum hold time value. You can adjust the hold time based on [Table 1-125](#).

Table 1-125 Mapping between the total number of BGP peers in all address families and the recommended minimum hold time

Total Number of Peers	Recommended Minimum Hold Time
0–100	20s
101–200	30s
201–300	45s
301–400	60s
401–500	75s
Greater than or equal to 501	90s

Avoid the following situations when setting values for the three timers:

- The *keepalive-time* and *hold-time* values are both set to 0. In this case, BGP timers become invalid, and BGP will not send Keepalive messages.
- The *hold-time* value is much greater than the *keepalive-time* value, for example, *keepalive-time* is set to 1 and *hold-time* is set to 65535. If the hold time is too long, BGP cannot detect the validity of the connection in time.
- The *keepalive-time* value is set to 0. In this case, the keepalive timer does not start. As a result, the *send-hold-time* function does not take effect.

After a connection is established between peers, the *keepalive-time* and *hold-time* values are negotiated by the peers. The smaller of the *hold-time* values carried in the Open messages exchanged between the peers is used as the final *hold-time* value. The smaller of one third of the negotiated *hold-time* value and the locally configured *keepalive-time* value is used as the final *keepalive-time*.

d. Run **commit**

The configuration is committed.

- Configure timers for a specific peer or peer group.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run the **peer { ipv4-address | group-name } timer keepalive keepalive-time hold hold-time [min-holdtime min-hold-value]** command to set the interval at which Keepalive messages are sent and the hold time for a peer or peer group. Alternatively, run the **peer ipv4-address timer send-hold send-hold-time** command to set the hold time during which the local device does not proactively disconnect from the peer end.

The relationship between the Keepalive and hold timer values in this case is the same as that in the scenario where global timers are configured.

 NOTE

You are advised to configure *hold-time* according to the total number of peers in all BGP address families. The more peers, the larger the recommended minimum hold time value. You can adjust the hold time based on [Table 1-126](#).

Table 1-126 Mapping between the total number of BGP peers in all address families and the recommended minimum hold time

Total Number of Peers	Recommended Minimum Hold Time
0–100	20s
101–200	30s
201–300	45s
301–400	60s
401–500	75s
Greater than or equal to 501	90s

d. Run **commit**

The configuration is committed.

----End

Configuring a MinRouteAdvertisementIntervalTimer

A proper MinRouteAdvertisementIntervalTimer can be configured to suppress frequent route changes, improving BGP network stability.

Context

BGP peers use update messages to exchange routing information. Update messages can be used to advertise reachable routes with the same attributes or delete unreachable routes.

BGP does not periodically update a routing table. When BGP routes change, BGP updates the changed BGP routes in the BGP routing table by sending Update messages. If a route changes frequently, to prevent the router from sending Update messages upon every change, configure a MinRouteAdvertisementIntervalTimer at which Update messages are sent.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **peer { ipv4-address | group-name } route-update-interval interval**

A MinRouteAdvertisementIntervalTimer is configured.

ipv4-address specifies the address of a specific peer, while *group-name* specifies the name of a peer group. The MinRouteAdvertisementIntervalTimer configured for a peer takes precedence over that configured for a peer group.

Step 4 Run **commit**

The configuration is committed.

----End

Disabling Rapid EBGP Connection Reset

Disabling rapid EBGP connection reset can prevent frequent reestablishment and deletion of EBGP sessions if route flapping occurs. This speeds up BGP network convergence.

Context

With rapid EBGP connection reset, BGP can immediately respond to a fault on an interface and delete the direct EBGP sessions on the interface without waiting for the hold timer to expire, which speeds up BGP network convergence. Rapid EBGP connection reset is enabled by default.

 NOTE

Rapid EBGP connection reset enables BGP to quickly respond to interface faults, but not interface recovery. After the interface recovers, BGP uses its state machine to restore relevant sessions.

If the status of an interface used to establish an EBGP connection changes frequently, the EBGP session will be deleted and reestablished repeatedly, causing network flapping. To address this issue, disable rapid EBGP connection reset so that BGP will not delete direct EBGP sessions on the interface until the hold timer expires. Therefore, disabling rapid EBGP connection reset suppresses BGP network flapping, speed up BGP network convergence, and reduce network bandwidth consumption.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **undo ebgp-interface-sensitive**

Rapid EBGP connection reset is disabled.

 NOTE

Disable rapid EBGP connection reset only when the status of an interface used to establish an EBGP connection changes frequently. If the status of the interface becomes stable, run the **ebgp-interface-sensitive** command to enable rapid EBGP connection reset.

Step 4 Run **commit**

The configuration is committed.

----End

Configuring a BGP ConnectRetry Timer

You can control the speed at which BGP peer relationships are established by changing the BGP ConnectRetry timer value.

Context

After BGP initiates a TCP connection, the ConnectRetry timer will be stopped if the TCP connection is established successfully. If the first attempt to establish a TCP connection fails, BGP re-establishes the TCP connection after the ConnectRetry timer expires.

- Setting a short ConnectRetry interval reduces the period BGP waits between attempts to establish a TCP connection, which speeds up the establishment of the TCP connection.
- Setting a long ConnectRetry interval suppresses routing flapping caused by frequent peer relationship flapping.

A ConnectRetry timer can be configured either for all peers or peer groups, or for a specific peer or peer group. A ConnectRetry timer configured for a specific peer takes precedence over that configured for the peer group of this peer. In addition, a ConnectRetry timer configured for a specific peer or peer group takes precedence over that configured for all peers or peer groups.

Procedure

- Configure a BGP ConnectRetry timer for all peers or peer groups.
Perform the following steps on a BGP router:
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **timer connect-retry connect-retry-time**
The ConnectRetry timer is configured for all BGP peers or peer groups.
 - d. Run **commit**
The configuration is committed.
- Configure a BGP ConnectRetry timer for a peer or peer group.
Perform the following steps on a BGP router:
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **peer { group-name | ipv4-address } timer connect-retry connect-retry-time**
The ConnectRetry timer is configured for a peer or peer group.
 - d. Run **commit**
The configuration is committed.

----End

Enabling Slow Peer Detection

After slow peer detection is configured on a device, the device identifies the slow BGP peer (if any) and removes it from the update peer-group to prevent this slow peer from affecting route advertisement to other peers in this update peer-group. Slow peer detection speeds up BGP network convergence.

Context

An update peer-group may consist of multiple peers. If a network problem (congestion for example) occurs and slows down the speed at which the local device advertises routes to a peer in the update peer-group, the speed at which

the local device advertises routes to other peers in the update peer-group is affected. Slow peer detection is enabled by default so that route advertisement to other peers in the update peer-group is not affected.

After slow peer detection is enabled, the local device calculates the difference between the time taken to send messages to each peer in the update peer-group and the shortest time taken to send messages to each peer in the group. If the difference between the time taken to send messages to a peer and the shortest time is greater than the threshold specified for slow peer detection, the local device considers this peer as a slow peer and removes it from the update peer-group, which prevents this slow peer from affecting route advertisement to other peers in the group.

If BGP Update messages fail to be advertised, slow peers cannot be detected using the original slow peer detection mode. To address this problem, configure the absolute mode for slow peer detection. If the delay in sending messages to a peer is greater than the absolute time threshold, the peer is considered a slow peer.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp *as-number***

The BGP view is displayed.

Step 3 Run **slow-peer detection threshold *threshold-value***

A threshold is configured for slow peer detection.

threshold *threshold-value* specifies a slow peer detection threshold. If the difference between the time taken to send messages to a peer in an update peer-group and the shortest time taken to send messages to each peer in the group is greater than the *threshold-value*, the local device considers this peer as a slow peer and removes it from the update peer-group.

Step 4 (Optional) Run **slow-peer absolute-detection detection threshold *absolute-threshold-value***

Slow peer detection in absolute time mode is enabled.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration of the BGP Network Convergence Speed

After the BGP network convergence speed is adjusted, you can view information about BGP peers and peer groups.

Prerequisites

Adjusting the BGP network convergence speed has been configured.

Procedure

- Run the **display bgp peer [verbose]** command to check information about BGP peers.
- Run the **display bgp group [group-name]** command to check information about BGP peer groups.
- Run the **display bgp slow-peer** command to check information about slow BGP peers.

----End

1.1.9.2.19 Configuring a Dynamic BGP Peer Group

Configuring dynamic BGP peer groups reduces network maintenance workload.

Usage Scenario

On a BGP network, multiple peers may frequently change, causing the establishment of peer relationships to change accordingly. If you configure peers in static mode, you need to frequently add or delete peer configurations on the local device, which increases the maintenance workload. To address this problem, configure the dynamic BGP peer function to enable BGP to listen for BGP connection requests from a specified network segment, dynamically establish BGP peer relationships, and add these peers to the same dynamic peer group. This spares you from adding or deleting BGP peer configurations in response to each change in BGP peers.

Pre-configuration Tasks

Before configuring a dynamic BGP peer group, complete the following task:

- **Configure basic BGP functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 (Optional) Run **bgp dynamic-session-limit max-num**

The maximum number of dynamic BGP peer sessions that can be established is set.

Step 4 Run **group group-name listen [internal | external | confederation-external]**

A dynamic BGP peer group is created.

Step 5 Run either of the following commands to configure the peer AS number or AS range from which the dynamic EBGP peer group listens for BGP connection requests.

- To specify the peer AS number from which the dynamic EBGP peer group listens for BGP connection requests, run the **peer group-name listen-as { asn } &<1-6>** command.
- To specify the peer AS range from which the dynamic EBGP peer group listens for BGP connection requests, run the **peer group-name listen-as-segment begin-as begin-asn end-as end-asn** command.

Step 6 Run **peer group-name listen-net ipv4-address { mask-length | mask }**

A network segment from which the dynamic BGP peer group listens for BGP connection requests is specified.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify the configuration.

- Run the **display bgp group [group-name]** command to check information about the dynamic BGP peer group.
- Run the **display bgp peer** command to check information about dynamic peers.

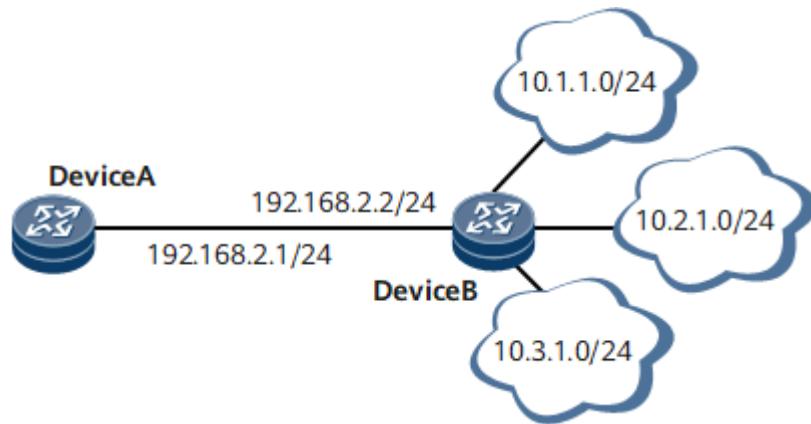
1.1.9.2.20 Configuring a BGP Device to Send a Default Route to Its Peer

After a BGP device is configured to send a default route to its peer, the BGP device sends a default route with the local address as the next hop address to the specified peer, regardless of whether there are default routes in the local routing table, which reduces the number of routes on the network.

Usage Scenario

The BGP routing table of the router on a medium or large BGP network contains a large number of routing entries. Storing the routing table consumes a large number of memory resources, and transmitting and processing routing information consume lots of network resources. If a device needs to send multiple routes to its peer, you can configure the device to send only a default route with the local address as the next hop address to its peer, regardless of whether there are default routes in the local routing table. This greatly reduces the number of routes on the network and the consumption of memory resources on the peer and network resources.

Figure 1-386 Configuring a device to send a default route to its peer



On the network shown in [Figure 1-386](#), Device A and Device B have established a BGP peer relationship. Device B has added routes to 10.1.1.0/24, 10.2.1.0/24, and 10.3.1.0/24 to its BGP routing table. Device A needs to learn these routes from Device B. In this way, Device A retains three BGP routes. To reduce the memory consumption on Device A and bandwidth used by Device B for sending routing information to Device A, configure Device B to send a default route to its peer (Device A) and use a routing policy to prevent Device B from sending all the routes to 10.1.1.0/24, 10.2.1.0/24, and 10.3.1.0/24 to Device A. Then, Device A stores only one default route but can still send traffic to the three network segments.

Pre-configuration Tasks

Before configuring a BGP device to send a default route to its peer, [configure basic BGP functions](#).

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run bgp *as-number*

The BGP view is displayed.

Step 3 Run ipv4-family unicast

The IPv4 unicast address family view is displayed.

Step 4 Run peer { *group-name* | *ipv4-address* } default-route-advertise [*route-policy route-policy-name* | *route-filter route-filter-name*] [*conditional-route-match-all* { *ipv4-address1* { *mask1* | *mask-length1* } } &<1-4> | *conditional-route-match-any* { *ipv4-address2* { *mask2* | *mask-length2* } } &<1-4>]

The device is configured to send default routes to the specified peer or a peer group.

You can specify **route-policy *route-policy-name*** or **route-filter *route-filter-name*** to change attributes of the default routes to be advertised.

If **conditional-route-match-all** { *ipv4-address1* { *mask1* | *mask-length1* } } &<1-4> is set, the BGP device sends default routes to the peer only when routes that match all the specified conditions exist in the local IP routing table.

If **conditional-route-match-any** { *ipv4-address2* { *mask2* | *mask-length2* } } &<1-4> is set, the local device sends default routes to the peer when routes that match any of the specified conditions exist in the local IP routing table.

NOTE

After the **peer default-route-advertise** command is used on a device, the device sends a default route with the local address as the next-hop address to a specified peer, regardless of whether there is a default route in the routing table.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, you can run the **display bgp routing-table [*ipv4-address* [*mask* | *mask-length*]]** command on the peer to view the received BGP default route.

1.1.9.2.21 Configuring a Device to Advertise BGP Supernet Unicast Routes to BGP Peers

This section describes how to configure a Border Gateway Protocol (BGP) device to advertise BGP supernet unicast routes to BGP peers.

Usage Scenario

A BGP supernet route has the same destination address and next hop address or has a more detailed destination address than the next hop address. Any route that meets one of the following conditions is a BGP supernet route.

- If bitwise AND operations are performed on the destination address mask with the destination address and next hop address, the two obtained network addresses are the same, and destination address mask is greater than or equal to the next hop address mask.
- If bitwise AND operations are performed on the destination address mask with the destination address and next hop address, the two obtained network addresses are different. If bitwise AND operations are performed on the next hop address mask with the destination address and next hop address, however, the two obtained network addresses are the same.

For example, the route with the destination address being 6.6.6.6 in the following command output is a BGP supernet route.

```
<HUAWEI> display bgp routing-table
BGP Local router ID is 1.1.1.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 1						
Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn	
*>i 6.6.6.6/32	6.6.6.6	0	100	0	i	

By default, when a BGP device receives a BGP supernet unicast route, the BGP device sets the route invalid and does not advertise it to other BGP peers. If a Huawei device is connected to a non-Huawei device and you want the Huawei device to advertise BGP supernet unicast routes that it receives from the non-Huawei device to other BGP peers, configure the Huawei device to advertise BGP supernet unicast routes to BGP peers.

Pre-configuration Tasks

Before you configure a BGP device to send BGP supernet unicast routes to BGP peers, [configure basic BGP functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

Step 4 Run **supernet unicast advertise enable**

The BGP device is enabled to advertise BGP supernet unicast routes to BGP peers.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After you configure a BGP device to advertise BGP supernet unicast routes, check whether the configuration takes effect.

- Run the **display bgp routing-table** command to check BGP supernet unicast routes.
- Run the **display bgp routing-table network** command to check information about a specified BGP supernet unicast route advertised to BGP peers.

1.1.9.2.22 Configuring BGP Load Balancing

BGP load balancing improves network resource usage and reduces network congestion.

Usage Scenario

On large networks, there may be multiple valid routes to the same destination. BGP, however, advertises only the optimal route to its peers, which may result in load imbalance.

Either of the following methods can be used to resolve the traffic imbalance:

- Use BGP routing policies to allow traffic to be balanced. For example, use a route-policy to modify the Local_Pref, AS_Path, Origin, or MED attribute of BGP routes to control traffic forwarding paths, helping implement load balancing.
- Use multiple paths to implement traffic load balancing. This method requires that multiple equal-cost routes exist and the number of routes allowed to participate in load balancing be set. Load balancing can be implemented globally or for a specified peer or peer group.

NOTE

- You can change load balancing rules through configurations. For example, you can prevent the device from comparing AS_Path attributes or IGP costs. When performing these configurations, ensure that no routing loops will occur.
- Locally leaked routes and routes imported between public network and VPN instances do not support load balancing.

Pre-configuration Tasks

Before configuring BGP load balancing, [configure basic BGP functions](#).

Procedure

- Configure BGP peer or peer group-based load balancing.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family unicast**
The BGP-IPv4 unicast address family view is displayed.
 - d. Run **peer { ipv4-address | group-name } load-balancing [as-path-ignore | as-path-relax]**
BGP peer or peer group-based load balancing is enabled.
- After the **peer load-balancing** command is run, load balancing is implemented only when the following conditions are met:
- The routes are received from a specified peer or peer group.
 - The routes are optimal routes or optimal equal-cost routes.
 - The AS_Path attribute of the routes is the same as that of the optimal route, or **as-path-ignore** or **as-path-relax** is specified when peer-based load balancing is configured.
 - If **as-path-ignore** is specified, the device ignores comparing AS_Path attributes when selecting routes for load balancing. In

- this case, routes can participate in load balancing even if their AS_Path attributes are different.
- If **as-path-relax** is specified, the device ignores comparing the AS_Path attributes of the same length when selecting routes for load balancing. In this case, routes cannot participate in load balancing if their AS_Path attributes are of different lengths. For example, the AS_Path attribute of route A is 10, and the AS_Path attribute of route B is 10, 20. Because the two AS_Path attributes are of different lengths, the two routes cannot participate in load balancing.
- e. (Optional) Enable peer or peer group-based load balancing among VPN routes.
- i. Run **quit**
Return to the BGP view.
 - ii. Run **quit**
Return to the system view.
 - iii. Run **ip vpn-instance *vpn-instance-name***
The VPN instance view is displayed.
 - iv. Run **ipv4-family unicast**
The VPN instance IPv4 address family view is displayed.
 - v. Run **route-distinguisher *route-distinguisher***
An RD is configured for the VPN instance IPv4 address family.
 - vi. Run **quit**
Return to the VPN instance view.
 - vii. Run **quit**
Return to the system view.
 - viii. Run **bgp *as-number***
The BGP view is displayed.
 - ix. Run **vpn-instance *vpn-instance-name***
A VPN instance is specified.
 - x. Run **peer { *ipv4-address* | *group-name* } as-number *as-number***
A peer relationship is established with the peer with the specified AS number.
 - xi. Run **quit**
Return to the BGP view.
 - xii. Run **ipv4-labeled-unicast vpn-instance *vpn-instance-name***
The BGP labeled VPN instance IPv4 address family view is displayed.
 - xiii. Run **peer { *ipv4-address* | *group-name* } enable**
The device is enabled to exchange routing information with the specified peer.
 - xiv. Run **peer { *ipv4-address* | *group-name* } load-balancing [**as-path-ignore** | **as-path-relax**]**
Peer or peer group-based load balancing among VPN routes is enabled.

f. (Optional) Change load balancing rules.

- To prevent the device from comparing AS_Path attributes when selecting routes for load balancing, run the **load-balancing as-path-ignore** command.
- To configure the device to ignore comparing the AS_Path attributes of the same length when selecting routes for load balancing, run the **load-balancing as-path-relax** command.
- To prevent the device from comparing IGP costs when selecting routes for load balancing, run the **load-balancing igr-metric-ignore** command.

 NOTE

The address family views supported by the preceding commands are different. When running any of the commands, ensure that the command is run in the correct address family view.

Change load balancing rules based on network requirements and exercise caution when running the commands.

g. Run **commit**

The configuration is committed.

- Configure global BGP load balancing.
 - Set the maximum number of BGP routes for load balancing.
 - i. Run **system-view**
The system view is displayed.
 - ii. Run **bgp as-number**
The BGP view is displayed.
 - iii. Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - iv. Run **maximum load-balancing [ebgp | ibgp] number [ecmp-nexthop-changed]**
The maximum number of equal-cost BGP routes for load balancing is set.
 - **ebgp** indicates that load balancing is implemented only among EBGP routes.
 - **ibgp** indicates that load balancing is implemented only among IBGP routes.
 - If neither **ebgp** nor **ibgp** is specified, both EBGP and IBGP routes can balance traffic, and the number of EBGP routes for load balancing is the same as the number of IBGP routes for load balancing.

 NOTE

Before routes to the same destination implement load balancing on a public network, a device determines the type of optimal route. If IBGP routes are optimal, only IBGP routes carry out load balancing. If EBGP routes are optimal, only EBGP routes carry out load balancing. This means that load balancing cannot be implemented using both IBGP and EBGP routes with the same destination address.

- v. (Optional) Change load balancing rules.
 - o To prevent the device from comparing AS_Path attributes when selecting routes for load balancing, run the **load-balancing as-path-ignore** command.
 - o To configure the device to ignore comparing the AS_Path attributes of the same length when selecting routes for load balancing, run the **load-balancing as-path-relax** command.
 - o To prevent the device from comparing IGP costs when selecting routes for load balancing, run the **load-balancing igr-metric-ignore** command.

 NOTE

The address family views supported by the preceding commands are different. When running any of the commands, ensure that the command is run in the correct address family view.

Change load balancing rules based on network requirements and exercise caution when running the commands.

- vi. Run **commit**

The configuration is committed.

- Set the maximum number of EBGP and IBGP routes for load balancing. This configuration is used in a VPN where a CE is dual-homed to two PEs. When the CE resides in the same AS as only one of the PEs, you can set the maximum number of EBGP and IBGP routes for load balancing so that VPN traffic can be balanced among EBGP and IBGP routes.

- i. Run **system-view**

The system view is displayed.

- ii. Run **bgp as-number**

The BGP view is displayed.

- iii. Run **ipv4-family vpn-instance vpn-instance-name**

The BGP-VPN instance IPv4 address family view is displayed.

- iv. Run **maximum load-balancing eibgp number [ecmp-nexthop-changed]**

The maximum number of EBGP and IBGP routes for load balancing is set.

After the **maximum load-balancing eibgp number** command is run on a device, the device, by default, changes the next hop of each route to itself before advertising the route to a peer, regardless of whether the route is to be used for load balancing. However, in RR or BGP confederation scenarios, the device does not change the next hop addresses of non-local routes to be advertised to a local address. As a result, besides the routes for load-balancing, those routes that are not supposed to participate in load balancing deliver traffic to the device, which overburdens the device. To address this problem, you can set **ecmp-nexthop-changed** so that the device changes the next hop of only routes that are to be used for load balancing to itself before advertising them to peers.

- v. (Optional) Change load balancing rules.

- To prevent the device from comparing AS_Path attributes when selecting routes for load balancing, run the **load-balancing as-path-ignore** command.
- To configure the device to ignore comparing the AS_Path attributes of the same length when selecting routes for load balancing, run the **load-balancing as-path-relax** command.
- To prevent the device from comparing IGP costs when selecting routes for load balancing, run the **load-balancing igr-metric-ignore** command.

 **NOTE**

The address family views supported by the preceding commands are different. When running any of the commands, ensure that the command is run in the correct address family view.

Change load balancing rules based on network requirements and exercise caution when running the commands.

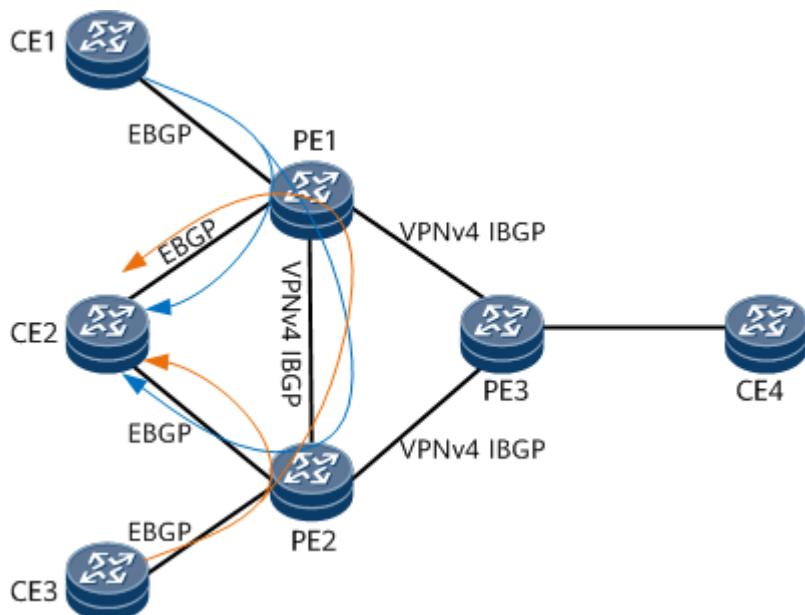
vi. Run **commit**

The configuration is committed.

- Configure load balancing among VPN unicast routes and leaked routes.

This configuration is mainly used in an EIBGP load balancing scenario where a CE that is single-homed to a PE accesses a CE that is dual-homed to PEs. As shown in [Figure 1-387](#), CE2 is dual-homed to PE1 and PE2. CE1 and CE2 are connected to PE1, and CE2 and CE3 are connected to PE2. When CE1 or CE3 needs to access CE2, you can configure load balancing among VPN unicast routes and leaked routes on PE1 and PE2. In this manner, when CE1 or CE3 accesses CE2, load balancing can be implemented through PE1 and PE2, that is, load balancing among VPN routes and leaked routes.

Figure 1-387 Load balancing among VPN unicast routes and leaked routes



- i. Run **system-view**
The system view is displayed.
- ii. Run **ip vpn-instance *vpn-instance-name***
A VPN instance is created, and its view is displayed.
- iii. Run **route-distinguisher *route-distinguisher***
An RD is configured for the VPN instance.
- iv. Run **apply-label { per-nexthop | per-route } pop-go**
A label distribution mode is configured for the current VPN.
- v. Run **quit**
Return to the system view.
- vi. Run **bgp *as-number***
The BGP view is displayed.
- vii. Run **ipv4-family vpn-instance *vpn-instance-name***
The BGP-VPN instance IPv4 address family view is displayed.
- viii. Run **load-balancing local-learning cross**
Load balancing among VPN unicast routes and leaked routes is configured.
The **load-balancing local-learning cross** command can be run only after the **apply-label { per-nexthop | per-route } pop-go** command is run in the corresponding address family view of a VPN instance. If the **apply-label { per-nexthop | per-route } pop-go** command is not run, routing loops may occur between PEs.
The **load-balancing local-learning cross** command is mutually exclusive with the **segment-routing ipv6 locator evpn**, **segment-routing ipv6 locator vxlan vni**, and **evpn mpls routing-enable** commands.
- ix. Run **commit**
The configuration is committed.

----End

Follow-up Procedure

When BGP routes carrying the link bandwidth extended community attribute are available for load balancing and they all recurse to IP routes or tunnels, you can run the **load-balancing ucmp** command in the BGP-IPv4 unicast address family view or BGP view to implement unequal-cost load balancing among BGP routes based on the link bandwidth extended community attribute. With this function, when there are multiple egress devices to the destination, unequal-cost load balancing can be implemented based on the actual bandwidth capability of each egress device. The methods of configuring the link bandwidth extended community attribute are as follows:

- Use **XPL to configure the link bandwidth extended community attribute**.
- Run the **peer generate-link-bandwidth** command to configure the local device to obtain the link bandwidth of a specified directly connected EBGP peer and generate the extended community attribute accordingly.

- Run the **peer generate-link-bandwidth** command to configure the local device to obtain the link bandwidth of a specified directly connected EBGP peer and generate the extended community attribute accordingly.

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp routing-table [network] [mask | mask-length] [longer-prefixes]** command to check information about the BGP routing table.
- Run the **display ip routing-table [verbose]** command to check information about the IP routing table.

1.1.9.2.23 Configuring BGP LSP Load Balancing

Configuring BGP LSP load balancing improves network resource utilization and reduces network congestion.

Context

On large networks, there may be multiple valid routes to the same destination. BGP, however, advertises only the optimal route to its peers. This may result in unbalanced traffic on different routes.

Either of the following methods can be used to resolve the problem:

- Use BGP policies for load balancing. For example, use a routing policy to modify the Local_Pref, AS_Path, Origin, or MED attribute of BGP routes to divert traffic to different forwarding paths for load balancing.
- Use multiple paths for load balancing. To use this method, equal-cost routes must exist and you need to configure the maximum number of load-balancing routes.

In some BGP LSP scenarios, for example, in the scenario where BGP unicast routes recurse to LSPs, or in the scenario where BGP LSPs can recurse to multiple TE/LDP tunnels, traffic must be balanced to prevent network congestion. BGP LSP load balancing allows traffic to be balanced based on the maximum number of load-balancing routes configured on ingress and transit nodes of BGP LSPs when the traffic is forwarded along the BGP LSPs, which improves network resource utilization and reduces network congestion.

Pre-configuration Tasks

Before configuring BGP LSP load balancing, establish BGP LSPs.

Procedure

- Configure ingress LSP load balancing on an ingress node.
 - Run **system-view**
The system view is displayed.
 - Run **bgp as-number**
The BGP view is displayed.

- c. (Optional) Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - d. Run **maximum load-balancing ingress-lsp *ingressNumber***
The maximum number of equal-cost BGP labeled routes is set for ingress LSP load balancing.
 - e. Run **commit**
The configuration is committed.
- Configure transit LSP load balancing on a transit node.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp *as-number***
The BGP view is displayed.
 - c. (Optional) Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - d. Run **maximum load-balancing transit-lsp *transitNumber***
The maximum number of equal-cost BGP labeled routes is set for transit LSP load balancing.
 - e. Run **commit**
The configuration is committed.

----End

Verifying the Configuration

After configuring transit LSP load balancing, you can run the **display bgp routing-table** and **display ip routing-table** commands to check whether load balancing has taken effect on the transit node. If the same destination IP address corresponds to more than one next hop, load balancing has taken effect.

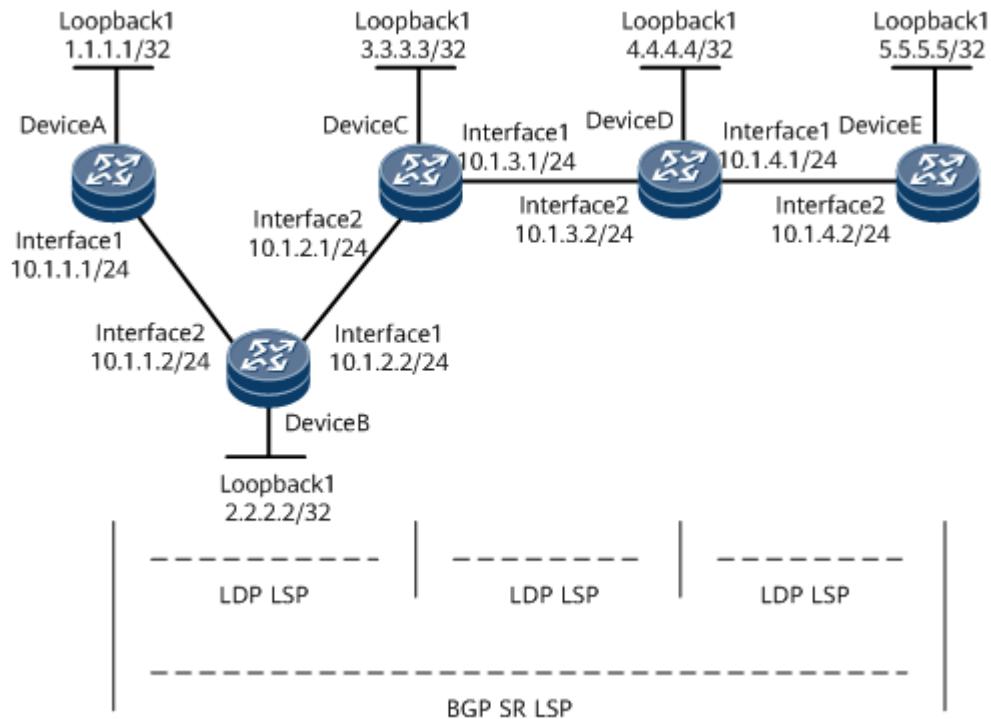
1.1.9.2.24 Configuring a BGP SR LSP

Deploying a complete BGP SR LSP on devices in the same AS helps implement end-to-end service interworking.

Context

On the network shown in [Figure 1-388](#), OSPF runs between DeviceA and DeviceB, between DeviceB and DeviceC, and between DeviceD and DeviceE. IS-IS runs between DeviceC and DeviceD. Basic MPLS capabilities and MPLS LDP are configured on DeviceA through DeviceE so that LDP LSPs are established between loopback interfaces of devices in each IGP area. Therefore, traffic between loopback interfaces of devices in each IGP area is encapsulated using MPLS. However, traffic cannot be transmitted across IGP areas because devices cannot ping each other across IGP areas. For example, DeviceA cannot ping DeviceE. To solve this problem, you need to configure an inner MPLS tunnel (BGP SR LSP) from 1.1.1.1 to 5.5.5.5 so that the traffic from 1.1.1.1 to 5.5.5.5 is forwarded through MPLS.

Figure 1-388 Configuring a BGP SR LSP



Procedure

- Configure an SRGB.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
BGP is started (with the local AS number specified), and the BGP view is displayed.
 - c. Run **segment-routing global-block begin-value end-value**
A BGP SRGB is configured.
 - d. Run **commit**
The configuration is committed.
- Configure a BGP peer relationship.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **peer ipv4-address as-number as-number**
The IP address of a peer and the number of the AS where the peer resides are specified.
 - d. Run **peer ipv4-address connect-interface interface-type interface-number [ipv4-source-address]**
A source interface and a source address to set up a TCP connection with the BGP peer are specified.

- e. Run **commit**
The configuration is committed.
- Configure DeviceC and DeviceD as RRs.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family unicast**
The BGP-IPv4 unicast address family view is displayed.
 - d. Run **peer { ipv4-address | group-name } reflect-client**
An RR is configured, and its peer is specified as a client.
Configure DeviceA and DeviceD as clients of DeviceC, and configure DeviceC and DeviceE as clients of DeviceD.
 - e. Run **peer { ipv4-address | group-name } next-hop-local**
The device is configured to set the next hop address to its own address when advertising routes to clients.
To enable DeviceC or DeviceD to change the next hop address of a route to the local address before advertising the route to clients, run the **peer next-hop-local** command on DeviceC or DeviceD for each related client.
 - f. Run **commit**
The configuration is committed.
- Enable the function to exchange labeled IPv4 routes.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **peer { ipv4-address | group-name } label-route-capability [check-tunnel-reachable]**
The device is configured to exchange labeled IPv4 routes with a specified BGP peer.
 - d. Run **commit**
The configuration is committed.
- Configure the ingress for a BGP SR LSP.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **route-policy route-policy-name matchMode node node**
A route-policy with a node is created, and the route-policy view is displayed.
 - c. Run **apply mpls-label**
The device is configured to allocate labels to IPv4 routes.
 - d. Run **quit**
Return to the system view.

- e. Run **bgp as-number**
The BGP view is displayed.
 - f. Run **network ipv4-address [mask | mask-length] [route-policy route-policy-name | route-filter route-filter-name] [non-relay-tunnel] label-index label-index-value**
BGP is configured to import a local route, and an offset is specified for the SRGB.
 - g. Run **peer { ipv4-address | group-name } route-policy route-policy-name export**
The route-policy is applied to the routes to be advertised to the specified BGP peer.
 - h. Run **ipv4-family unicast**
The BGP-IPv4 unicast address family view is displayed.
 - i. Run **peer peerIpv4Addr prefix-sid**
The device is configured to exchange prefix SID information with the specified IPv4 peer.
 - j. Run **commit**
The configuration is committed.
- Configure a transit node for the BGP SR LSP.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **route-policy route-policy-name matchMode node node**
A route-policy with a node is created, and the route-policy view is displayed.
 - c. Run **if-match mpls-label**
Labeled IPv4 routes are matched.
 - d. Run **apply mpls-label**
The device is configured to allocate labels to IPv4 routes.
 - e. Run **quit**
Return to the system view.
 - f. Run **bgp as-number**
The BGP view is displayed.
 - g. Run **peer { ipv4-address | group-name } route-policy route-policy-name export**
The route-policy is applied to the routes to be advertised to the specified BGP peer.
 - h. Run **ipv4-family unicast**
The BGP-IPv4 unicast address family view is displayed.
 - i. Run **peer peerIpv4Addr prefix-sid**
The device is configured to exchange prefix SID information with the specified IPv4 peer.
 - j. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp routing-table ipv4-address [mask | mask-length] prefix-sid srgb** command to check SRGB information of the BGP routes with a specified destination address.
- Run the **display mpls lsp** command to check BGP SR LSP information.

1.1.9.2.25 Configuring Path MTU Auto Discovery

Path MTU auto discovery allows BGP to discover the smallest MTU value on a path so that BGP messages are transmitted based on the path MTU. This function improves transmission efficiency and BGP performance.

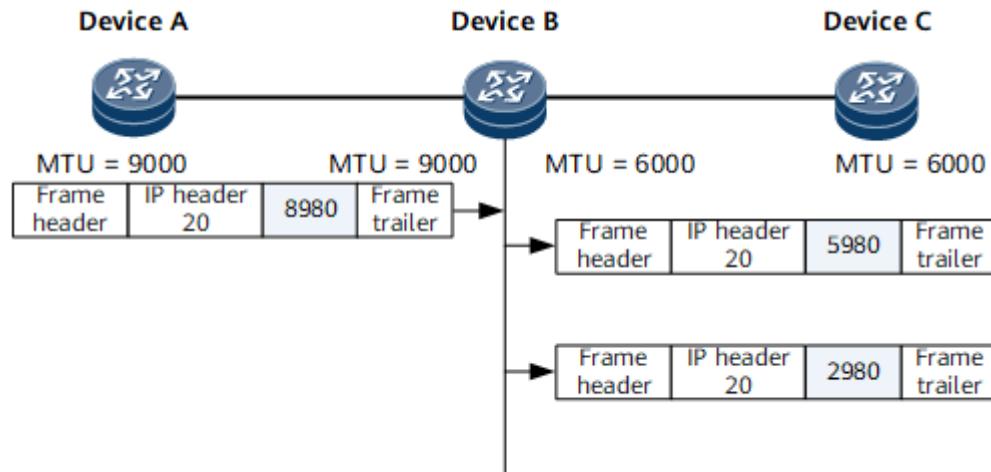
Usage Scenario

The link-layer MTUs of different networks that a communication path traverses may vary. The smallest MTU on the path is the most important factor that influences the communication between the two ends of the path. The smallest MTU on the communication path is called the path MTU.

The path MTU varies with the selected path and therefore may change. In addition, the path MTU in one direction may be inconsistent with that in the reverse direction. Enabling path MTU auto discovery allows a device to discover the path MTU from the transmit end to the receive end. TCP encapsulates IP packets based on the path MTU when transmitting BGP messages.

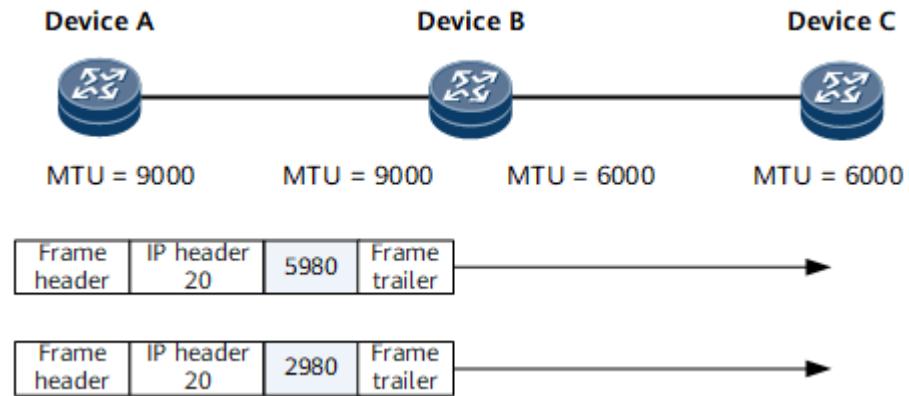
In [Figure 1-389](#), path MTU auto discovery is not enabled, and a BGP peer relationship is established between Device A and Device C. After Device A sends 9000-byte BGP messages to Device C, Device B fragments the messages upon reception by adding one IP header, one Layer 2 frame header, and one Layer 2 frame trailer in each fragment, which reduces transmission efficiency. If a fragment of a message is lost, the message becomes invalid.

Figure 1-389 Network without path MTU auto discovery



In [Figure 1-390](#), path MTU auto discovery is enabled, and a BGP peer relationship is established between Device A and Device C. Device A sends 9000-byte BGP messages to Device C, and the path MTU is 6000 bytes. In this case, Device B does not fragment the messages upon reception, which improves transmission efficiency.

Figure 1-390 Network with path MTU auto discovery



NOTE

Enabling path MTU auto discovery affects TCP MSS calculation.

- When path MTU auto discovery is not enabled:
 - For the sender, the TCP MSS is calculated using the following formula: **MSS = MIN { CFGMSS, MTU-40 }**
 - For the receiver:

When the device supports SYN Cookie, the MSS is calculated using the following formula: **MSS = MIN { MIN { CFGMSS, MTU-40 } , internally-defined MSS value }**

If the device does not support SYN Cookie, the MSS is calculated using the following formula: **MSS = MIN { CFGMSS, MTU-40 }**
- When path MTU auto discovery is enabled:
 - The sender updates the local MSS only when it sends a packet with the MSS greater than the path MTU. The TCP MSS is calculated using the following formula: **MSS = MIN { MIN { CFGMSS, MTU-40 } , PMTU-40 }**
 - For the receiver:

If the device supports SYN Cookie, the TCP MSS is calculated using the following formula: **MSS = MIN { MIN { MIN { CFGMSS, MTU-40 } , internally-defined MSS value } , PMTU-40 }**

If the device does not support SYN Cookie, the TCP MSS is calculated using the following formula: **MSS = MIN { MIN { CFGMSS, MTU-40 } , PMTU-40 }**

Parameters in the formula are described as follows:

- CFGMSS: **MIN { APPMSS, CLICFGMSS }**
 - APPMSS: MSS value configured using the **peer tcp-mss** command
 - CLICFGMSS: maximum MSS value configured using the **tcp max-mss mss-value** command
- MTU-40: interface MTU minus 40
- PMTU-40: path MTU minus 40
- internally-defined MSS value: MSS value, including 216, 460, 952, 1400, 2900, 4900, 7900, and 9500. Upon receipt of a packet, the receive-end device uses the internally-defined MSS value which is smaller than but closest to the MSS of the received packet.

Pre-configuration Tasks

Before configuring path MTU auto discovery, [configure basic BGP functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **peer { group-name | ipv4-address } path-mtu auto-discovery**

Path MTU auto discovery is enabled.

After the command is run, a BGP peer learns the path MTU, preventing BGP messages from being fragmented during transmission.

 NOTE

A BGP message from one end to the other may travel a path different from the path used by the ACK message that is responded by the other end. Therefore, running this command on both ends is recommended so that both peers exchange messages based on the path MTU.

Step 4 Run **quit**

Return to the system view.

Step 5 Run **tcp timer pathmtu-age age-time**

The aging time is set for an IPv4 path MTU.

The path MTUs vary with the path. If there are multiple routes between two communication hosts and the routes selected for packet transmission change frequently, configure the path MTU aging time so that the system updates path MTUs based on the path MTU aging time, increasing the transmission efficiency.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After performing all configurations related to path MTU auto discovery, you can run the **display bgp peer [ipv4-address] verbose** command to check whether path MTU auto discovery is successfully configured in the detailed BGP peer information.

1.1.9.2.26 Configuring BGP Route Recursion to the Default Route

When the next hop of a BGP route is not directly reachable, you can configure BGP route recursion to the default route.

Usage Scenario

The next hops of BGP routes may not be directly reachable. In this case, recursion is required so that the BGP routes can be used for traffic forwarding. You can configure whether to allow BGP routes to recurse to the default route.

Pre-configuration Tasks

Before configuring BGP route recursion to the default route, [configure basic BGP functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run `ipv4-family unicast`

The BGP-IPv4 unicast address family view is displayed.

Step 4 Run `nexthop recursive-lookup default-route`

BGP route recursion to the default route is configured.

Step 5 Run `commit`

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the `display current-configuration` command in the system view to check whether BGP route recursion to the default route is configured.

1.1.9.2.27 Configuring BGP Next Hop Recursion Based on a Route-Policy

Configuring BGP next hop recursion based on a route-policy prevents traffic loss if routes changes.

Usage Scenario

When BGP routes change, BGP needs to perform route recursion on the BGP routes with indirect next hops. If no route-policies are configured to filter the routes on which a BGP route with an indirect next hop depends for recursion, the BGP route may recurse to an incorrect route, which may cause traffic loss. To address this problem, configure BGP next hop recursion based on a route-policy. If no routes match the route-policy, the BGP route with an indirect next hop is considered unreachable. In this situation, incorrect route recursion and traffic loss are prevented.

Pre-configuration Tasks

Before configuring BGP next hop recursion based on a route-policy, complete the following tasks:

- [Configure basic BGP functions](#).
- [Configure a route-policy](#).

NOTICE

Before configuring a route-policy, ensure that the routes on which BGP routes with indirect next hops depend for recursion will not be filtered out by the route-policy. Otherwise, route recursion fails, and traffic cannot be forwarded.

Procedure

Step 1 Run `system-view`

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **nexthop recursive-lookup { route-policy route-policy-name | route-filter route-filter-name }**

BGP next hop recursion based on a route-policy or route-filter is configured.



The command does not apply to the routes received from directly connected EBGP peers or LinkLocal peers.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display bgp routing-table network [mask | mask-length]** command to check detailed information about a specified route in the BGP routing table.

1.1.9.2.28 Configuring AIGP value on a Route-Policy

BGP prefers the route with the smallest AIGP value during BGP route selection.

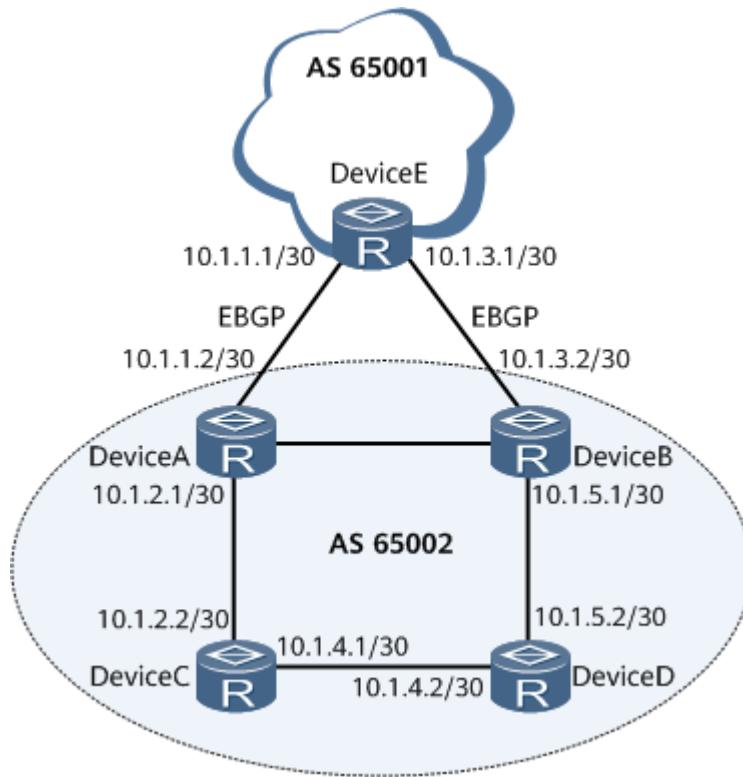
The Accumulated Interior Gateway Protocol Metric (AIGP) attribute is an optional non-transitive Border Gateway Protocol (BGP) path attribute. After the AIGP attribute is configured in an AIGP administrative domain, BGP selects paths based on costs in the same manner as an IGP, and all devices in the domain forward data along the optimal routes. During BGP route selection, the AIGP attribute is used as follows:

- The priority of a route that carries the AIGP attribute is higher than the priority of a route that does not carry the AIGP attribute.
- If two BGP routes both carry the AIGP attribute, the device selects the BGP route whose AIGP value plus the cost of the IGP route to which the BGP route recurses is smaller.

The AIGP attribute can be added to routes only through route-policies. You can configure an **apply** clause for a route-policy using the **apply aigp { [+ | -] cost | inherit-cost }** command to modify the AIGP value during route import, acceptance, or advertisement by BGP. If no AIGP value is configured, the IGP routes imported by BGP do not carry the AIGP attribute.

Figure 1-391 is used as an example to show how the AIGP attribute is used during BGP route selection. In **Figure 1-391**, OSPF runs in AS 65002, and an EBGP peer relationship is established between DeviceA and DeviceE, and between DeviceB and DeviceE. BGP is configured on DeviceA and DeviceB to import OSPF routes in AS 65002 and advertise the routes to AS 65001.

Figure 1-391 AIGP application networking



Run the **display bgp routing-table [ip-address]** command on Device E to check the configurations. The route 10.1.4.0/30 is used in this example.

Display the routing table of Device E.

```
[DeviceE] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete

Total Number of Routes: 6
      Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
      *> 10.1.2.0/30    10.1.1.2      0        0       65002??
      *     10.1.3.2    3            0        0       65002??
      *> 10.1.4.0/30    10.1.1.2      2        0       65002??
      *     10.1.3.2    2            0        0       65002??
      *> 10.1.5.0/30    10.1.3.2      0        0       65002??
      *     10.1.1.2    3            0        0       65002??

[DeviceE] display bgp routing-table 10.1.4.0
BGP local router ID : 10.1.1.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.1.2 (10.1.1.2)
Route Duration: 00h02m29s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
  10.1.1.2
  10.1.3.2
BGP routing table entry information of 10.1.4.0/30:
```

```
From: 10.1.3.2 (10.1.5.1)
Route Duration: 00h03m58s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, pre 255, not preferred for router ID
Not advertised to any peer yet
```

The command output when the AIGP attribute is not configured shows that DeviceE selects the route learned from DeviceA after comparing router IDs. To change the route selection result on DeviceE, you can configure the AIGP attribute.

Configurations on Device A:

```
#  
bgp 65002  
#  
ipv4-family unicast  
import-route ospf 1 route-policy aigp_policy //Apply route-policy named aigp_policy to locally  
imported OSPF routes and use aigp_policy to modify the AIGP value.  
peer 10.1.1.1 aigp //Enable AIGP on the local device and the peer 10.1.1.1.  
#  
route-policy aigp_policy permit node 10 //Define the first node of aigp_policy and set the AIGP  
value of the route 10.1.4.0/30 to 10.  
if-match ip-prefix prefix1  
apply aigp 10  
#  
route-policy aigp_policy permit node 20 //Define the second node of aigp_policy and allow  
aigp_policy to permit all routes.  
#  
ip ip-prefix prefix1 index 10 permit 10.1.4.0 30 //Configure IP prefix list named prefix1 to match the  
route 10.1.4.0/30.  
#
```

Configurations on Device B:

```
bgp 65002  
peer 10.1.3.1 as-number 65001  
#  
ipv4-family unicast  
import-route ospf 1 route-policy aigp_policy1 //Apply route-policy named aigp_policy1 to locally  
imported OSPF routes and use aigp_policy1 to modify the AIGP value.  
peer 10.1.3.1 aigp //Enable AIGP on the local device and the peer 10.1.3.1.  
#  
route-policy aigp_policy1 permit node 10 //Define the first node of aigp_policy1 and set the AIGP  
value of the route 10.1.4.0/30 to 5.  
if-match ip-prefix prefix2  
apply aigp 5  
#  
route-policy aigp_policy1 permit node 20 //Define the second node of aigp_policy1 and allow  
aigp_policy1 to permit all routes.  
#  
ip ip-prefix prefix2 index 10 permit 10.1.4.0 30 //Configure IP prefix list named prefix2 to match the  
route 10.1.4.0/30.  
#
```

Configurations on Device E:

```
#  
bgp 65001  
#  
ipv4-family unicast  
peer 10.1.1.2 aigp //Enable AIGP on the local device and the peer 10.1.1.2.  
peer 10.1.3.2 aigp //Enable AIGP on the local device and the peer 10.1.3.2.  
#
```

Run the **display bgp routing-table [ip-address]** command on Device E to check the configurations.

Display the routing table of Device E.

```
[DeviceE] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete

Total Number of Routes: 6
Network          NextHop      MED     LocPrf   PrefVal Path/Ogn
*> 10.1.2.0/30    10.1.1.2    0        0       65002?
*   10.1.3.2      3           0        65002?
*> 10.1.4.0/30    10.1.3.2    2        0       65002?
*   10.1.1.2      2           0        65002?
*> 10.1.5.0/30    10.1.3.2    0        0       65002?
*   10.1.1.2      3           0        65002?

[DeviceE] display bgp routing-table 10.1.4.0
BGP local router ID : 10.1.1.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.3.2 (10.1.5.1)
Route Duration: 00h00m14s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, best, select, active, pre 255, AIGP 5
Advertised to such 2 peers:
  10.1.1.2
  10.1.3.2
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.1.2 (10.1.1.2)
Route Duration: 01h01m15s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, pre 255, AIGP 10, not preferred for AIGP
Not advertised to any peer yet
```

The preceding command output shows that Device E selects the route 10.1.4.0/30 learned from Device B because its AIGP value is smaller.

Table 1-127 shows the attribute comparison of the routes learned from DeviceA and DeviceB.

Table 1-127 Attribute comparison of the routes learned from Device A and Device B.

Route Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
PrefVal	0	0	The same.
Local_Pref	-	-	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	10	5	The different.

1.1.9.2.29 Configuring BGP Add-Path

BGP Add-Path allows a device to send multiple routes with the same prefix to a specified IBGP peer. These routes can back up each other or load-balance traffic, which improves network reliability.

Usage Scenario

In a scenario with an RR and clients, if the RR has multiple routes to the same destination (with the same prefix), the RR selects an optimal route from these routes and then sends only the optimal route to its clients. Therefore, the clients have only one route to the destination. If a link along this route fails, route convergence takes a long time, which cannot meet the requirements for high reliability.

To address this issue, deploy the BGP Add-Path feature on the RR. With BGP Add-Path, the RR can send two or more routes with the same prefix to a specified IBGP peer. These routes can back up each other or load-balance traffic, which ensures high reliability in data transmission. The BGP Add-Path feature does not affect BGP route selection rules.

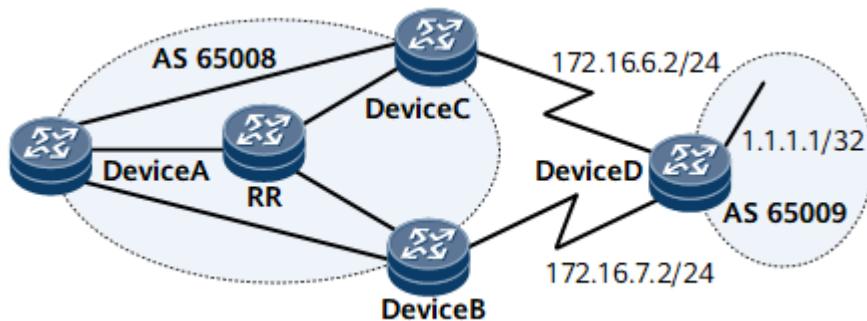
NOTE

BGP Add-Path is not supported in BGP confederation scenarios.

Add-Path takes effect only for routes learned from peers, not for local routes.

Generally, BGP Add-Path is configured on an RR, and the Add-Path route receiver needs to be configured to accept such routes. In [Figure 1-392](#), you can enable BGP Add-Path on the RR, enable Device A to accept BGP Add-Path routes from the RR so that Device A obtains two routes destined for 1.1.1.1/32, with next hops of 172.16.6.2 and 172.16.7.2. The two routes can back up each other or load-balance traffic.

Figure 1-392 Networking for configuring BGP Add-Path



Pre-configuration Tasks

Before configuring BGP Add-Path, [configure basic BGP functions](#).

Procedure

- Perform the following steps on the RR:
 - a. Run **system-view**

- The system view is displayed.
- b. (Optional) Run **route-policy** *route-policy-name* **matchMode** **node** *node*
A route-policy is created, and the route-policy view is displayed.
- c. (Optional) Run **quit**
Return to the system view.
- d. (Optional) Run **xpl route-filter** *route-filter-name*
A route-filter is created, and the route-filter view is displayed.
- e. (Optional) Run **end-filter**
The system view is displayed.
- f. Run **bgp** *as-number*
The BGP view is displayed.
- g. Run **peer** { *ipv4-address* | *ipv6-address* | *peerGroupName* } **as-number** *as-number*
An IP address and AS number are specified for a peer.
- h. (Optional) Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
- i. Run **bestroute add-path path-number** *path-number*
BGP Add-Path is enabled, and the number of Add-Path routes is configured.
- j. Run **peer** { *ipv4-address* | *group-name* } **capability-advertise add-path send**
The device is enabled to send Add-Path routes to the specified peer.
- k. Run **peer** { *peerIpv4Addr* | *groupName* } **advertise add-path path-number** *number* { **route-policy** *route-policy-name* | **route-filter** *route-filter-name* }
The maximum number of preferred routes to be advertised to a specified peer is specified.

NOTE

To configure **route-policy** *route-policy-name*, you need to enter the route-policy view.

To configure **route-filter** *route-filter-name*, you need to enter the route-filter view.

- l. Run **commit**
The configuration is committed.
- Perform the following steps on Device A:
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp** *as-number*
The BGP view is displayed.

- c. Run **peer { ipv4-address | group-name } capability-advertise add-path receive**

The device is enabled to accept Add-Path routes from the specified peer.

- d. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display bgp peer verbose** command to check the BGP Add-Path status.

1.1.9.2.30 Configuring the POPGO Function

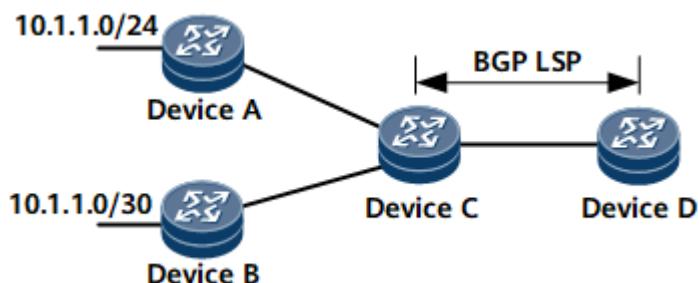
After the POPGO function is configured on the egress of a BGP LSP, the egress forwards each data packet received from the LSP through the outbound interface found in the ILM based on the label information carried in the packet.

Usage Scenario

On the network shown in [Figure 1-393](#), DeviceC has two static routes, 10.1.1.0/24 and 10.1.1.0/30, and the next hops of the two routes are DeviceA and DeviceB, respectively. DeviceC only imports route 10.1.1.0/24 to BGP and then sends this route to DeviceD. A BGP LSP is established between DeviceC and DeviceD. By default, after DeviceC receives a data packet destined for 10.1.1.0 from DeviceD, DeviceC removes the BGP LSP label from the packet, searches the IP forwarding table for the outbound interface, and forwards the packet along the route 10.1.1.0/30 based on the longest match rule. That is, the packet is incorrectly forwarded to DeviceB.

To solve the preceding problem, run the **apply-label per-route pop-go** command in the BGP view on DeviceC. After the **apply-label per-route pop-go** command is configured, when DeviceC sends route 10.1.1.0/24 to DeviceD, DeviceC records in the ILM the mapping between the label assigned to the route and the outbound interface of the route. Then, after the DeviceC receives a data packet from DeviceD, DeviceC directly searches the ILM for an outbound interface based on label information carried in the packet and forwards the packet through the found outbound interface after removing the packet label. This implementation ensures correct packet forwarding.

[Figure 1-393](#) POPGO networking



Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **apply-label per-route pop-go**

The POPGO function is configured.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

Run the **display mpls lsp protocol bgp** command on the device to view detailed information about BGP LSP establishment. If **POPGO** is displayed in the **Label Operation** field, the POPGO function is successfully configured.

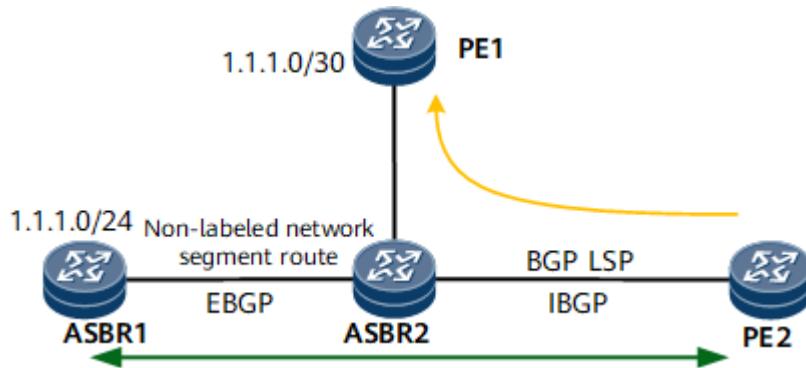
1.1.9.2.31 Configuring the Device to Perform the Label Pop-go Action for Packets Carrying the Label Added to Each Received Non-labeled Unicast Route

Configure the labeled route sender to perform the label pop-go action for packets if the packets carry the labels allocated to non-labeled routes received in the BGP-IPv4 unicast address family, so that traffic can be forwarded through the specified path when the destination of the traffic is reachable through an IP route, but not through an LSP.

Usage Scenario

On the network shown in [Figure 1-394](#), an IBGP peer relationship is established between PE2 and ASBR2, and an EBGP peer relationship is established between ASBR1 and ASBR2. ASBR1 is reachable to ASBR2 through an IP route, but not through the LSP between the two ASBRs. After receiving the packets destined for ASBR1 from PE2, ASBR2 sends the packets to PE1 according to the longest match rule. To allow ASBR2 to perform the pop-go action for the BGP label in the received packets destined for ASBR1 and search the local ILM for an outbound interface based on the label to send the packets to ASBR1, run the **unicast-route label pop-go** command.

Figure 1-394 Usage scenario of the **unicast-route label pop-go** command



Procedure

- Perform the following operations on ASBR2:
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family labeled-unicast**
The BGP-labeled address family view is displayed.
 - d. Run **import-rib { public | vpn-instance vpn-instance-name } [valid-route] [route-policy route-policy-name | route-filter route-filter-name]**
The device is configured to import public network routes to the labeled address family.

NOTE

To specify **vpn-instance vpn-instance-name**, you need to create the VPN instance.

- e. Run **unicast-route label pop-go**

The device is configured to perform the label pop-go action for packets if the packets carry the labels allocated to non-labeled routes received in the BGP-IPv4 unicast address family.

If the destination of received packets is reachable through an IP route, but not through an LSP, the device pops out the labels that are carried in received packets and allocated to non-labeled routes received in the BGP-IPv4 unicast address family and forwards the packets based on the next hop and outbound interface information.

- f. Run **quit**
Return to the system view.
- g. Run **commit**
The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display bgp routing-table label [statistics]** command to check information about labeled routes in the BGP routing table.

1.1.9.2.32 Configuring conversion from BGP IPv4 Unicast Routes to Labeled Routes

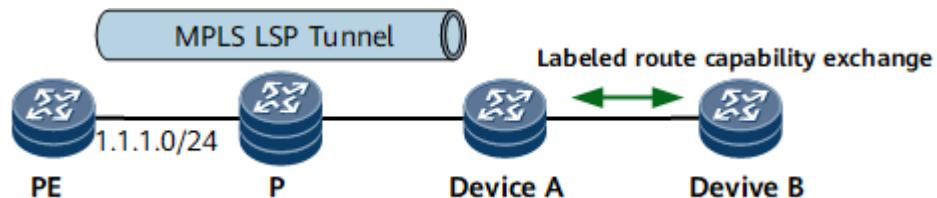
Configuring conversion from BGP IPv4 unicast routes to labeled routes can ensure that traffic is forwarded along a specified LSP.

Usage Scenario

On the network shown in [Figure 1-395](#), an IBGP peer relationship and an MPLS LSP are established between the user device and DeviceA. It is required that traffic from DeviceB to the user device be forwarded along the MPLS LSP.

To implement this function, enable DeviceA and DeviceB to send or receive labeled routes, and run the **unicast-route label advertise** command on DeviceA. After DeviceA assigns MPLS labels to routes based on a route-policy, DeviceA converts the IPv4 public network unicast route (1.1.1.0/24) learned from the user device to a labeled route and sends the labeled route to DeviceB. After traffic from DeviceB reaches DeviceA, DeviceA performs the POPGO action. Specifically, DeviceA removes the BGP label, adds an LSP label, and forwards the traffic to the user device along the MPLS LSP.

Figure 1-395 Networking for configuring conversion from BGP IPv4 unicast routes to labeled routes



Procedure

- Perform the following operations on the labeled route sender (DeviceA):
 - a. Run **system-view**
The system view is displayed.
 - b. Run **route-policy route-policy-name matchMode node node**
A route-policy to be used to filter the routes to be advertised is created.
 - c. Run **apply mpls-label**
The device is enabled to assign labels to IPv4 routes.
 - d. Run **quit**
The system view is displayed.

- e. Run **bgp as-number**

The BGP view is displayed.

- f. Run **peer { group-name | ipv4-address } label-route-capability [check-tunnel-reachable]**

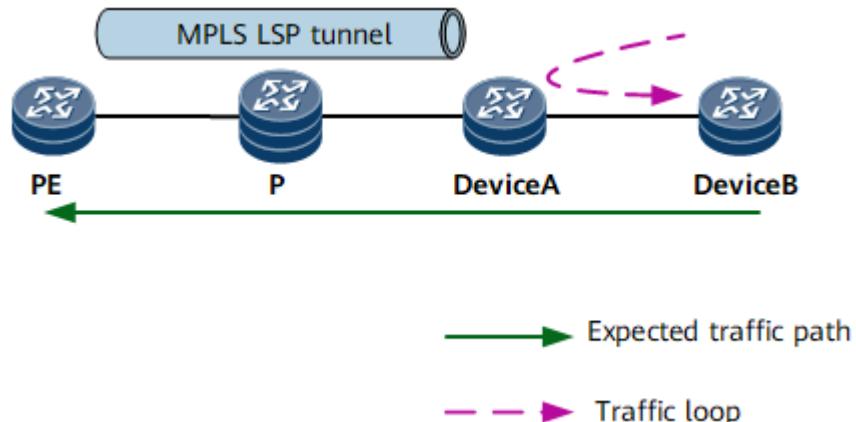
The device is enabled to send labeled routes to a specified peer or peer group.

 NOTE

To prevent a traffic loop, it is recommended that **check-tunnel-reachable** be specified to allow the device to check tunnel reachability.

Figure 1-396 shows the networking diagram for a traffic loop. If **check-tunnel-reachable** is not specified, DeviceA converts the routes learned from the user side into labeled routes and advertises them to DeviceB, regardless of whether the tunnel is reachable. Traffic from DeviceB is forwarded based on the BGP label. When the traffic reaches DeviceA, the BGP label is removed. If the LSP is unreachable, the traffic recurses to a specific outbound interface and next hop. In this case, if a route that DeviceA learns from another device is more specific than that learned from the user side, traffic will be forwarded over an incorrect route or even be routed back to DeviceB, thereby causing a traffic loop.

Figure 1-396 Networking diagram for a traffic loop



- g. Run **peer { group-name | ipv4-address } route-policy route-policy-name export**

The route-policy is configured as an export policy based on which the device advertises routes to the peer.

- h. Run **unicast-route label advertise**

The device is enabled to convert received IPv4 unicast routes to labeled routes and advertise the labeled routes to peers with the labeled route exchange capability.

- i. (Optional) Run **unicast-route label advertise pop-go**

The device is enabled to convert received IPv4 public network unicast routes into labeled routes and advertise the labeled routes to peers with the labeled route exchange capability.

- If the IP address of an outbound interface is reachable but no LSP is reachable, traffic is forwarded through the outbound interface and a specific next hop, during which the label POPGO action is performed.
- j. Run **quit**
Return to the system view.
 - k. Run **commit**
The configuration is committed.
 - Perform the following operations on the labeled route receiver (DeviceB):
 - a. Run **bgp as-number**
The BGP view is displayed.
 - b. Run **peer { group-name | ipv4-address } label-route-capability [check-tunnel-reachable]**
The device is enabled to receive labeled routes.
 - c. Run **quit**
Return to the system view.
 - d. Run **commit**
The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display bgp routing-table label [statistics]** command to check information about labeled routes in the BGP routing table.

1.1.9.2.33 Configuring BFD for BGP

BFD for BGP speeds up fault detection and therefore increases the route convergence speed.

Usage Scenario

Currently, voice and video services are widely applied. These services are quite sensitive to the packet loss and delay. BGP periodically sends Keepalive messages to a peer to monitor the peer's status, but this mechanism takes an excessively long time, more than 1 second, to detect a fault. If data is transmitted at Gbit/s rates and a link fault occurs, such a lengthy detection period will result in a large amount of data being lost, making it impossible to meet the high reliability requirements of carrier-grade networks.

To address this issue, BFD for BGP has been introduced. BFD for BGP detects faults on links between BGP peers within milliseconds. The fast detection speed ensures fast BGP route convergence and minimizes traffic loss.

Pre-configuration Tasks

Before configuring BFD for BGP, complete the following task:

- Configure parameters of the link layer protocol and IP addresses for interfaces to ensure that the link layer protocol on the interfaces is up.
- Configure basic BGP functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bfd**

BFD is enabled globally.

Step 3 Run **quit**

Return to the system view.

Step 4 Run **bgp as-number**

The BGP view is displayed.

Step 5 (Optional) Run **ipv4-family vpn-instance vpn-instance-name**

The BGP-VPN instance IPv4 address family view is displayed.



Using this command, you can enter the BGP-VPN instance IPv4 address family view so that BFD for BGP can be configured for the VPN instance in this view. To configure BFD for BGP for the public network, skip this step.

Step 6 Run **peer { group-name | ipv4-address } bfd enable [single-hop-prefer | { per-link one-arm-echo }] [compatible]**

BFD is configured for a peer or peer group, and default BFD parameters are used to establish a BFD session.

A BFD session can be established only when the peer relationship is in the Established state.

After BFD is enabled for a peer group, BFD sessions will be created on the peers that belong to this peer group and are not configured with the **peer bfd block** command.

The **compatible** parameter configures the compatibility mode, which ensures that the local and peer devices use the same detection mode when a Huawei device is connected to a non-Huawei device. The **compatible** parameter must be used together with the **single-hop-prefer** parameter.

- In the scenario where Huawei devices are interconnected, the same parameters must be configured on the local and peer devices. If the devices are configured with different BFD parameters, services may be affected.
- In the scenario where a Huawei device is connected to a non-Huawei device, to implement BFD interworking, select a proper configuration mode from the following table to ensure consistent configurations on the local and peer devices.

Table 1-128 Configuration modes for the **compatible** and **single-hop-prefer** parameters

Whether single-hop-prefer Is Configured	Whether compatible Is Configured	Direct IBGP Scenario	Multi-hop IBGP Scenario	Direct EBGP Scenario	Multi-hop EBGP Scenario
N	N	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 253.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 253.	In the packets sent by BFD, the UDP port number is 3784, and the TTL is 253.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 253.
Y	N	In the packets sent by BFD, the UDP port number is 3784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 253.	In the packets sent by BFD, the UDP port number is 3784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 253.
N	Y	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 3784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 255.

Whether single-hop-prefer Is Configured	Whether compatible Is Configured	Direct IBGP Scenario	Multi-hop IBGP Scenario	Direct EBGP Scenario	Multi-hop EBGP Scenario
Y	Y	In the packets sent by BFD, the UDP port number is 3784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 255. If the peer UDP port number is 3784, the local BFD session can learn the port number and change the local UDP port number to 3784.	In the packets sent by BFD, the UDP port number is 3784, and the TTL is 255.	In the packets sent by BFD, the UDP port number is 4784, and the TTL is 255. If the peer UDP port number is 3784, the local BFD session can learn the port number and change the local UDP port number to 3784.

 NOTE

In the preceding table, the UDP port numbers in the packets sent by BFD are those obtained during initial negotiation. BFD has the automatic adaptation function. If the UDP port number of a received response packet is different from that of the sent packet, BFD automatically changes the UDP port number of the sent packet to be the same as the received one and re-sends the packet to the peer end.

Step 7 (Optional) Run **peer { group-name | ipv4-address } bfd { min-tx-interval min-tx-interval | min-rx-interval min-rx-interval | detect-multiplier multiplier } ***

BFD session parameters are modified.

 NOTE

The BFD parameters of a single peer take precedence over those of a peer group. If BFD parameters are configured on peers, they will be used in BFD session establishment.

When changing the default values, pay attention to the network status and the network reliability requirement. A short interval for transmitting BFD packets can be configured for a link that has a higher reliability requirement. A long interval

for transmitting BFD packets can be configured for a link that has a lower reliability requirement.

 NOTE

There are three formulas: Actual interval for the local device to send BFD packets = max {Locally configured interval for transmitting BFD packets, Remotely configured interval for receiving BFD packets}, Actual interval for the local device to receive BFD packets = max {Remotely configured interval for transmitting BFD packets, Locally configured interval for receiving BFD packets}, and Local detection period = Actual interval for receiving BFD packets x Remotely configured BFD detection multiplier.

For example:

- On the local device, the configured interval for transmitting BFD packets is 200 ms, the interval for receiving BFD packets is 300 ms, and the detection multiplier is 4.
- On the peer device, the configured interval for transmitting BFD packets is 100 ms, the interval for receiving BFD packets is 600 ms, and the detection multiplier is 5.

Then:

- On the local device, the actual interval for transmitting BFD packets is 600 ms calculated by using the formula max {200 ms, 600 ms}; the interval for receiving BFD packets is 300 ms calculated by using the formula max {100 ms, 300 ms}; the detection period is 1500 ms calculated by multiplying 300 ms by 5.
- On the peer device, the actual interval for transmitting BFD packets is 300 ms calculated by using the formula max {100 ms, 300 ms}; the interval for receiving BFD packets is 600 ms calculated by using the formula max {200 ms, 600 ms}; the detection period is 2400 ms calculated by multiplying 600 ms by 4.

Step 8 (Optional) Run **peer { group-name | ipv4-address } bfd valid-ttl-hops valid-ttl-hops-value**

A TTL value is set for checking the BFD session with a specified peer or peer group.

In scenarios where EBGP peers are indirectly connected, you can configure a TTL value for a BFD session so that the traffic forwarding path is quickly adjusted if BFD detects a fault on the link between the local device and the specified BGP peer. Specifically, the local interface used to establish the BFD session forwards only the packets with a TTL value greater than or equal to the configured TTL value. If the TTL value in a BFD packet is smaller than the configured TTL value, the interface discards this packet, the BFD session goes down, and BFD notifies BGP of this event. In this manner, BGP routes are re-converged so that the traffic forwarding path is adjusted.

The **peer bfd valid-ttl-hops** command and the **peer bfd enable single-hop-prefer** command are mutually exclusive.

The **peer bfd valid-ttl-hops** command and the **peer bfd enable per-link one-arm-echo** command are mutually exclusive.

Step 9 (Optional) Run **peer ipv4-address bfd block**

A peer is prevented from inheriting the BFD function of the peer group to which it belongs.

If a peer joins a peer group enabled with BFD, the peer inherits the BFD configuration of the group and creates a BFD session. To prevent the peer from inheriting the BFD function of the peer group, perform this step.

 NOTE

The **peer bfd block** command and the **peer bfd enable** command are mutually exclusive.
After the **peer bfd block** command is run, the BFD session is automatically deleted.

Step 10 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After BFD for BGP is configured, you can run the **display bgp bfd session** {[**vpnv4 vpn-instance** *vpn-instance-name*] **peer** *ipv4-address* | **all**} command to view information about the BFD session established by BGP.

1.1.9.2.34 Configuring BGP Peer Tracking

BGP peer tracking provides fast link or peer fault detection for BGP to speed up network convergence.

Context

BFD can be configured to detect peer relationship status changes in order to implement rapid BGP convergence. BFD, however, needs to be configured on the entire network and has poor extensibility. If BFD cannot be deployed to detect BGP peer relationship status changes, you can configure BGP peer tracking on the local device to quickly detect link or peer unreachability, implementing rapid network convergence. In addition, you can adjust the interval between detecting peer unreachability and terminating the BGP connection to suppress BGP peer relationship flapping caused by route flapping, thereby improving BGP network stability.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp** *as-number*

The BGP view is displayed.

Step 3 Run **peer** { *group-name* | *ipv4-address* } **tracking** [**delay** *delay-time*]

BGP peer tracking is configured for a specified peer or peer group.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.9.2.35 Configuring BGP Auto FRR

BGP Auto FRR, a protection measure against link faults, applies to the network topology with both primary and backup links. It can be configured for services that are quite sensitive to the packet loss and delay.

Usage Scenario

As networks evolve continuously, voice, on-line video, and financial services raise increasingly high requirements for real-time performance. Usually, primary and backup links are deployed on a network to ensure the stability of these services. In a traditional forwarding mode, the router selects a route out of several routes that are bound for the same destination network as the optimal route and delivers the route to the FIB table to guide data forwarding. If the optimal route fails, the router has to wait for route convergence to be completed before reselecting an optimal route and delivering it to the FIB table. In this case, services are interrupted for a long time, unable to meet service requirements.

BGP Auto FRR allows a device to select the optimal route from the routes that are bound for the same destination network and automatically add information about the suboptimal route to the backup forwarding entry of the optimal route. If the primary link fails, the device quickly switches traffic to the backup link. The switchover does not depend on route convergence and can be performed within sub-seconds, greatly reducing service interruption time.

Pre-configuration Tasks

Before configuring BGP Auto FRR, complete the following tasks:

- [Configure basic BGP functions.](#)

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The BGP IPv4 unicast address family view is displayed.

Step 4 Perform either of the following operations to enable BGP Auto FRR for unicast routes:

- To enable only BGP Auto FRR for unicast routes, run the **auto-frr** command.
- To enable BGP Auto FRR for unicast routes and configure the function of preferentially selecting the SRv6 BE path of the primary route as the backup path, run the **auto-frr best-effort** command. If a device is configured to preferentially select the SRv6 BE path of the primary route as the backup path and both the SRv6 TE-Policy and SRv6 BE path of the primary route are available, the device preferentially selects the SRv6 BE path of the primary route as the backup path. That is, the SRv6 BE path of the primary route has a

higher priority than the backup route. If the SRv6 TE-Policy of the primary route fails, the forwarding path is rapidly switched to this SRv6 BE path.

Step 5 (Optional) Run **route-select delay delay-value**

Delayed route selection is configured. After the primary path recovers, delayed route selection ensures that the device on the primary path performs route selection only after the corresponding forwarding entries on the device are stable. This prevents traffic loss during traffic switchback.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring BGP Auto FRR, you can run the following commands to check the previous configuration.

- Run the **display bgp routing-table [network [{ mask | mask-length } [longer-prefixes]]]** command to check information in a BGP routing table.
- Run the **display ip routing-table [ip-address [mask | mask-length] [longer-match]] verbose** command to check backup forwarding entries in an IP routing table.

1.1.9.2.36 Configuring Delayed Response to BGP Next Hop Recursion Changes

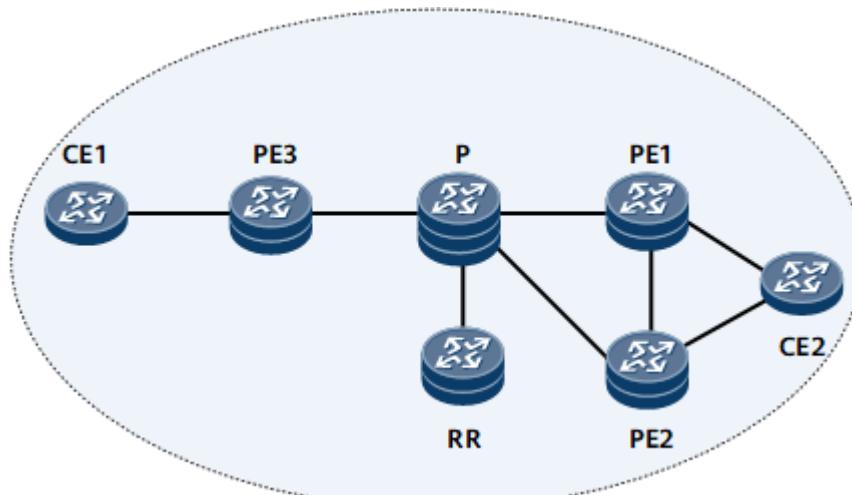
Configuring delayed response to BGP next hop recursion changes can minimize traffic loss during route changes.

Usage Scenario

As shown in [Figure 1-397](#), PE1, PE2, and PE3 are the clients of the RR. CE2 is dual-homed to PE1 and PE2. PE1 and PE2 advertise their routes destined for CE2 to the RR. The RR advertises the route from PE1 to PE3. PE3 has only one route to CE2 and advertises this route to CE1. After the route exchange, CE1 and CE2 can communicate. If PE1 fails, PE3 detects that the next hop is unreachable and instructs CE1 to delete the route to CE2. Traffic is interrupted. After BGP route convergence is complete, the RR selects the route advertised by PE2 and sends a route update message to PE3. PE3 then advertises this route to CE1, and traffic forwarding is restored. A high volume of traffic will be lost during traffic interruption because BGP route convergence is rather slow.

If delayed response to BGP next hop recursion changes is enabled on PE3, PE3 does not reselect a route or instruct CE1 to delete the corresponding route immediately after detecting that the route to PE1 is unreachable. After BGP convergence is complete, the RR selects the route advertised by PE2 and sends the route to PE3. PE3 then reselects a route and sends a route update message to CE1. Traffic forwarding is restored. After delayed response to BGP next hop recursion changes is enabled on PE3, PE3 does not need to delete the route or instruct CE1 to delete the route. This delayed response speeds up BGP route convergence and minimizes traffic loss.

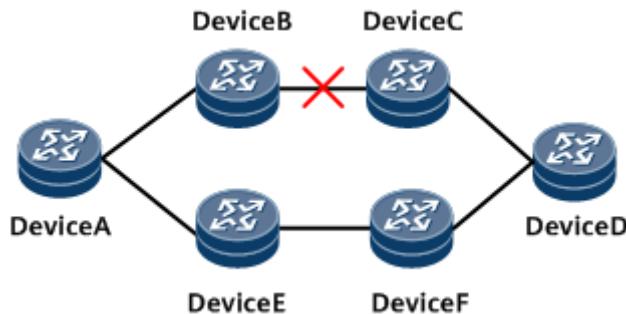
Figure 1-397 Networking diagram for configuring the BGP next hop recursion change delayed response



There are two links between DeviceA and DeviceD, and traffic passes through link DeviceA -> DeviceB -> DeviceC -> DeviceD.

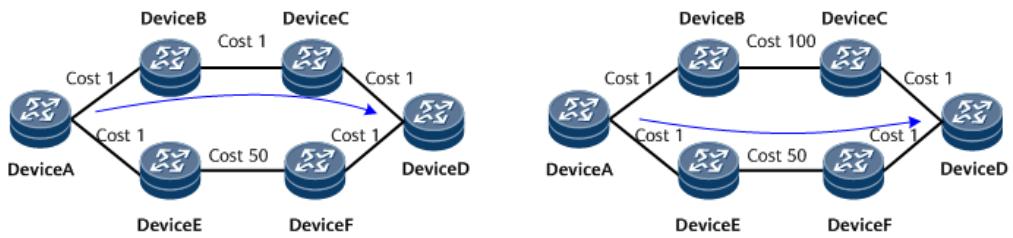
- On the network shown in [Figure 1-398](#), if the link between DeviceB and DeviceC fails, BGP cannot find the next hop route or tunnel for recursion to DeviceC. As a result, traffic is interrupted and is switched to the link DeviceA -> DeviceE -> DeviceF -> DeviceD. In this case, when the next hop recursion changes, the reachability also changes. This is called a critical recursion change.

Figure 1-398 Networking diagram for a critical recursion change



- On the network shown in [Figure 1-399](#), when the link between DeviceB and DeviceC is normal, the cost of this link is increased. Due to route selection, traffic is switched to the link DeviceA -> DeviceE -> DeviceF -> DeviceD. In this case, the next hop recursion result changes, but the reachability remains unchanged. This is called a non-critical recursion result change.

Figure 1-399 Networking diagram for a non-critical recursion change



NOTE

Delayed response to BGP next hop recursion changes applies only to scenarios where multiple links exist between the downstream device and the same destination. If there is only one link between the downstream device and the destination, configuring delayed response to BGP next hop recursion changes may cause heavier traffic loss when the link fails because link switching is impossible.

Pre-configuration Tasks

Before configuring the BGP next hop recursion change delayed response, complete the following task:

- [Configure basic BGP functions.](#)

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Configure delayed response to next hop recursion changes.

- To configure a delay in responding to next hop recursion changes, run the **nexthop recursive-lookup delay [delay-time] [include-tunnel]** command. After the command is run, the system delays to respond to both critical and non-critical next hop recursion changes.
- To configure a delay in responding to non-critical next hop recursion changes, run the **nexthop recursive-lookup non-critical-event delay [nonCrit-delay-time]** command.
- To configure a delay in responding to critical next hop recursion changes (from unreachability to reachability), run the **nexthop recursive-lookup critical-event-reachable delay [critReach-delay-time] [include-tunnel]** command.

These commands can be configured separately or together. If they are all configured, the first command has a lower priority than the **nexthop recursive-lookup non-critical-event delay [nonCrit-delay-time]** command or the **nexthop recursive-lookup critical-event-reachable delay [critReach-delay-time] [include-tunnel]** command.

Step 4 Run commit

The configuration is committed.

----End

Verifying the Configuration

After configuring delayed response to BGP next hop recursion changes, you can run the **display current-configuration configuration bgp | include nexthop recursive-lookup** command to view the delay in responding to BGP next hop recursion changes.

1.1.9.2.37 Setting a Specified BGP Peer or Each Peer in a Peer Group as an Independent Update Peer-Group

Setting a specified peer or each peer in a peer group as an independent update peer-group prevents routes learned from the peer from being sent back to the peer.

Usage Scenario

To improve the efficiency of route advertisement, BGP uses the dynamic update peer-group mechanism. The BGP peers with the same configurations are placed in an update peer-group. These routes are grouped once and then sent to all peers in the update peer-group, improving the grouping efficiency exponentially. However, the routes learned from a peer may be sent back to the peer, for example, the preferred route learned from an EBGP peer is sent back to the EBGP peer, or the preferred route that an RR learns from a client is reflected back to the client. In this case, messages are discarded, wasting network resources.

To address this problem, you can set a specified peer or each peer in a peer group as an independent update peer-group so that the routes learned from the peer are not sent back to the peer.

NOTE

Setting a specified peer or each peer in a peer group as an independent update peer-group can be performed in multiple address families. This section uses the IPv4 unicast address family as an example. For details about the address families supported, see **peer update-group-independent**.

Pre-configuration Tasks

Before configuring a specified peer or each peer in a peer group as an independent update peer-group, complete the following tasks:

- [Configure basic BGP functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run `ipv4-family unicast`

The IPv4 unicast address family view is displayed.

Step 4 Set a specified peer or each peer in a peer group as an independent update peer-group.

NOTE

The configuration of a peer takes precedence over that of the peer group to which the peer belongs.

- To set a specified peer as an independent update peer-group, run the **peer *ipv4-address* update-group-independent enable** command.
- To set each peer in a peer group as an independent update peer-group, run the **peer *group-name* update-group-independent** command.

Step 5 Run `commit`

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, you can run the **display bgp update-peer-group** command to view information about update peer-groups.

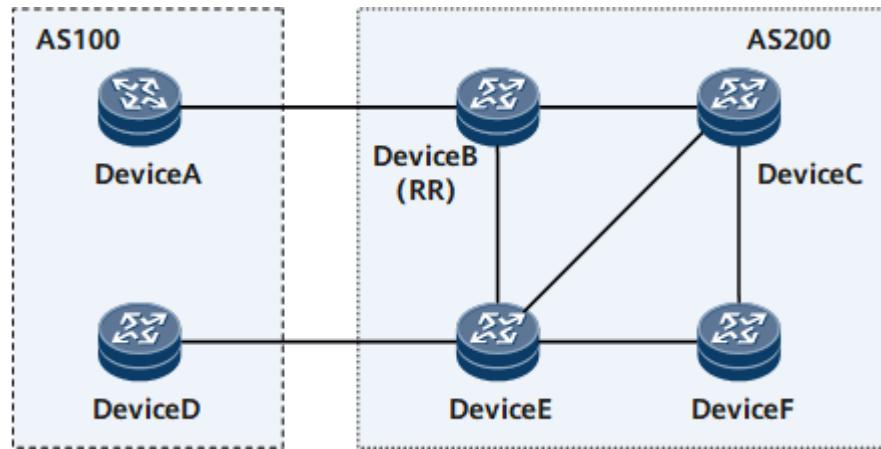
1.1.9.2.38 Configuring a Delay in Releasing Obtained Labels in a BGP LSP FRR Switchover Scenario

Configuring a delay in releasing obtained labels in a BGP LSP FRR switchover scenario can prevent second-time packet loss.

Usage Scenario

In a BGP LSP FRR switchover scenario shown in [Figure 1-400](#), DeviceA and DeviceD belong to AS 100, and DeviceB, DeviceC, DeviceE, and DeviceF belong to AS 200. The optimal route selected by DeviceB is a route learned from its EBGP peer DeviceA, and the suboptimal route selected by DeviceB is a route learned from its IBGP peer DeviceE. In normal cases, DeviceB preferentially selects the labeled BGP route learned from its EBGP peer DeviceA and sends the route to its IBGP peer DeviceC. DeviceB delivers an incoming label map (ILM) entry and next hop label forwarding entry (NHLFE). DeviceC also delivers an NHLFE. If DeviceA restarts due to a fault, DeviceB withdraws the route learned from DeviceA. The suboptimal route learned from its IBGP peer DeviceE becomes the new optimal route and is not sent to its IBGP peer DeviceC. Instead, DeviceB (RR) reflects this route to DeviceC. Therefore, DeviceB releases the label that has been applied for, deletes the ILM entry. If the NHLFE entry of DeviceC is updated slowly, traffic is still sent to DeviceB. In this case, packet loss occurs again because the ILM entry of DeviceB has been deleted.

Figure 1-400 BGP LSP FRR switchover networking



To prevent this problem, configure a delay in releasing obtained labels (deleting ILM entries) on Device B.

Pre-configuration Tasks

Before configuring a delay in releasing obtained labels in a BGP LSP FRR switchover scenario, complete the following tasks:

- [Configure basic BGP functions](#).
- [Configure BGP Auto FRR](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **label-free delay delay-value**

A delay in releasing obtained labels is set.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

Run the **display current-configuration** command to check the valid configuration.

1.1.9.2.39 Configuring the Route Server Function

This section describes how to configure the route server function. The function reduces network resource consumption on ASBRs.

Usage Scenario

In some scenarios on the live network, to achieve network traffic interworking, EBGP full-mesh connections may be required. However, establishing full-mesh connections among border devices is costly and places high requirements on the performance of the devices, which adversely affects the network topology and device expansion. The route server function is similar to the RR function used in IBGP full-mesh connection scenarios and allows devices to advertise routes to their clients (border devices) without changing route attributes, such as AS_Path, Nexthop, and MED. This reduces network resource consumption on the border devices.

Pre-configuration Tasks

Before configuring the route server function, complete the following tasks:

- [Configure basic BGP functions](#).

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 (Optional) Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **peer { ipv4-address | group-name } route-server-client**

The route server function is enabled on the device, and an EBGP peer is specified as its client.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring the route server function, verify the configuration.

Run the **display bgp routing-table** command to view routes in the BGP routing table.

1.1.9.2.40 Configuring the BGP GR Helper

You can configure a device to function as a Graceful Restart (GR) helper to help a BGP peer with the BGP GR process.

Usage Scenario

When BGP restarts, the peer relationship is re-established, and traffic forwarding is interrupted. To address this issue, enable GR.

GR takes effect only when BGP GR is enabled and a GR-capable BGP session is established between the GR restarter and its peers.

Pre-configuration Tasks

Before configuring the BGP GR helper, [configure basic BGP functions](#).

Enabling BGP GR

Enabling or disabling GR may delete and reestablish all sessions and instances.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **graceful-restart**

BGP GR is enabled.

Step 4 (Optional) Run **graceful-restart timer restart restart-time**

The maximum time is set for the local end to wait for GR recovery on the peer end.

Step 5 (Optional) Run **graceful-restart peer-reset**

The router is enabled to reset a BGP session in GR mode.

Currently, BGP does not support dynamic capability negotiation. Therefore, a BGP capability change causes the re-establishment of the peer relationship. In a scenario where a BGP IPv4 unicast peer relationship has been established and IPv4 services are running properly, if a BGP capability changes, the BGP IPv4 unicast peer relationship will be re-established, affecting IPv4 services. To prevent this problem, run the **graceful-restart peer-reset** command to enable the router to reset the BGP connection in GR mode.

Step 6 (Optional) Run **rpd-family**

The BGP RPD address family view is displayed.

Step 7 Run **peer { ipv4-address | ipv6-address } enable**

The device is enabled to exchange routing information with the specified peer.

Step 8 Run **peer *ipv4-address* graceful-restart static-timer *restart-time***

The maximum time is configured for the local end to wait for the peer end to recover from GR.

Step 9 Run **commit**

The configuration is committed.

----End

Configuring the Parameter for a BGP GR Session

Changing the parameter of a BGP GR session may re-establish BGP peer relationships.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp *as-number***

The BGP view is displayed.

Step 3 Run **graceful-restart timer wait-for-rib *time***

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the BGP GR Helper Configuration

After configuring a BGP GR helper, verify the BGP GR status.

Prerequisites

The BGP GR helper has been configured.

Procedure

- Run the **display bgp peer verbose** command to check the BGP GR status.
- Run the **display bgp graceful-restart status** command to check the GR information about a BGP speaker.

----End

1.1.9.2.41 Enabling GR for BGP Peers

After the GR capability is configured for a BGP peer, a BGP speaker can negotiate with the peer to establish a BGP session with the GR capability.

Usage Scenario

A BGP restart causes re-establishment of all the involved peer relationships, resulting in traffic interruption. Enabling GR globally can prevent traffic

interruption. However, this will disconnect all the BGP peer relationships on a device and cause the device to re-negotiate the GR capability with these peers, which affects all the BGP-dependent services running on the live network. To prevent this problem, you can enable GR on a BGP speaker for specified BGP peers so that the BGP speaker can establish GR-capable BGP sessions with the specified peers through negotiation.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **peer { ipv4-address | group-name } capability-advertise graceful-restart**

GR is enabled, and the device is configured to advertise the GR capability to the specified BGP peer or peer group.

NOTE

If a specified peer group contains peers that do not support GR, you can run the **peer group-name local-graceful-restart enable** command to enable local GR for the specified peer group.

Step 4 (Optional) Run **peer { ipv4-address | group-name } graceful-restart timer restart time-value**

The maximum duration is set for a specified peer or peer group to wait for the local BGP peer relationship to be re-established, and the device is configured to advertise the duration to the peer or peer group.

NOTE

If the specified peer does not support GR, run the **peer ipv4-address local-graceful-restart timer restart** command to set the maximum duration for the device to wait for its BGP peer relationship to be re-established with the specified peer.

If a peer in a specified peer group does not support GR, you can run the **peer group-name local-graceful-restart timer restart** command to set the maximum duration for the local device to wait for its peer relationships to be re-established with the specified BGP peer group.

Step 5 (Optional) Run **peer { ipv4-address | group-name } graceful-restart peer-reset**

The device is enabled to use the GR mode to reset the BGP connection with the specified peer or each peer in the specified group.

Currently, BGP does not support dynamic capability negotiation. Therefore, each time a BGP capability is changed or a new BGP capability is enabled, a BGP speaker tears down the existing sessions with the affected peers and renegotiates BGP capabilities with these peers. For example, a BGP speaker has established a BGP IPv4 unicast peer relationship with a peer, and the IPv4 service is running properly. A change of the BGP capability causes the BGP IPv4 unicast peer relationship to be reestablished, affecting the normal running of the IPv4 service. To address this issue, run the **peer { ipv4-address | group-name } graceful-restart peer-reset** command.

Step 6 (Optional) Run **peer { ipv4-address | group-name } graceful-restart timer wait-for-rib time-value**

The maximum duration is set for the device to wait for the End-of-RIB flag from the specified peer.

 **NOTE**

If the specified peer does not support GR, you can run the **peer ipv4-address local-graceful-restart timer wait-for-rib wfrtime** command to set the duration for the local end to wait for the End-of-RIB flag from the specified peer.

If a specified peer group contains a peer that does not support GR, you can run the **peer group-name local-graceful-restart timer wait-for-rib wfrtime** command to set the duration for the local end to wait for the End-of-RIB flag from the specified peer group.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the following commands to verify the configuration.

- Run the **display bgp peer verbose** command to check the BGP GR status.
- Run the **display bgp graceful-restart status** command to check GR information on the BGP speaker.
- Run the **display bgp local-graceful-restart status** command to check information about local GR on the BGP speaker.

1.1.9.2.42 Configuring BGP G-Shut

After EBGP and IBGP sessions between ASBRs and BGP devices are shut down during maintenance, the ASBRs and BGP devices are temporarily unreachable to each other during BGP convergence. To minimize traffic loss during session closure and re-establishment, you can configure the g-shut function to gracefully shut down one or more BGP sessions.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Enable and activate BGP g-shut globally or for a specified peer as required.

- Enable and activate BGP g-shut globally.
 - a. Run the **graceful-shutdown all-peer** command to enable the g-shut function globally.
 - b. Run the **graceful-shutdown manual-activate** command to activate the g-shut function globally.

- Enable and activate BGP g-shut for a peer.
 - a. Run the **peer { peerIpv4Addr | peerIpv6Addr | groupName } graceful-shutdown [local-preference local-preference-value | as-prepend as-prepend-value]** command to enable the g-shut function for a peer or peer group.
 - b. Run the **peer { peerIpv4Addr | peerIpv6Addr | groupName } graceful-shutdown manual-activate** command to activate the g-shut function for a peer or peer group.

 NOTE

If a peer group has been configured with g-shut but peer A in it does not need to inherit g-shut from the peer group, run the **peer { peerIpv4Addr | peerIpv6Addr } graceful-shutdown disable** command. To prevent this peer from inheriting the g-shut activation status from the peer group, run the **peer { peerIpv4Addr | peerIpv6Addr } graceful-shutdown manual-activate disable** command.

Step 4 (Optional) Run **advertise-community-gshut { ibgp | ebpg }**

The device is configured to advertise the g-shut community attribute of the address family level.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the following commands to verify the configuration.

Run the **display bgp peer verbose** command to check the status of BGP g-shut.

1.1.9.2.43 Configuring BGP Best-external

Border Gateway Protocol (BGP) Best-external can speed up route convergence if the primary link fails.

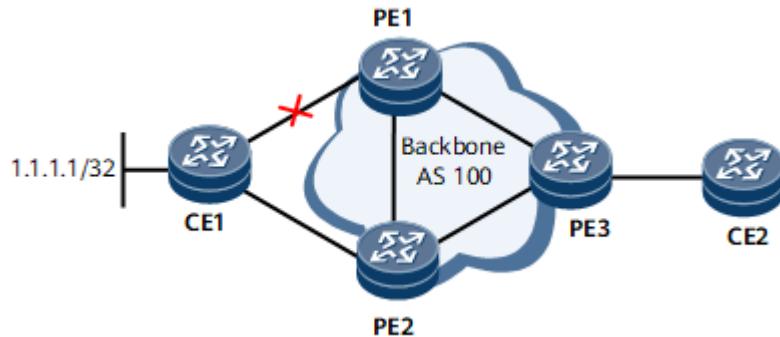
Usage Scenario

If multiple routes to the same destination are available, a BGP device selects one optimal route based on BGP route selection policies and advertises the route to its BGP peers. This optimal route may be advertised by either an External Border Gateway Protocol (EBGP) peer or an Internal Border Gateway Protocol (IBGP) peer.

However, in scenarios with master and backup provider edges (PEs), if routes are selected based on the preceding policies and the primary link fails, the BGP route convergence takes a long time because no backup route is available. To address this problem, configure BGP Best-external on the backup PE.

The following figure shows the networking with master and backup PEs.

Figure 1-401 Networking with master and backup PEs



BGP Best-external must be enabled on the backup PE (PE2).

Pre-configuration Tasks

Before configuring BGP Best-external, [configure basic BGP functions](#).

Configuration Procedures

1. Run **system-view**
The system view is displayed.
2. Run **bgp as-number**
The BGP view is displayed.
3. Run **bestroute best-external**
The device is enabled to select BGP Best-external routes.
4. Run **peer { ipv4-address | group-name } advertise best-external**
The device is enabled to advertise Best-external routes to the specified peer or peer group.
5. Run **commit**
The configuration is committed.

Verifying the Configuration

After the configuration is complete, run the following command to verify it.

Run the **display bgp peer verbose** command to check the BGP Best-external status.

1.1.9.2.44 Configuring BMP

BGP Monitoring Protocol (BMP) allows the BGP running status and route processing records and on network devices to be monitored in real time.

Usage Scenario

BMP is mainly used in networking scenarios where monitoring servers exist and need to monitor the BGP running status and route processing records and on network devices in real time. The running status includes the establishment and

disconnection of peer relationships and update of routing information. BGP route processing records indicate the process of processing BGP routes on a device, for example, processing of the routes that match an import or export policy. Without BMP, only manual query can be used to obtain the BGP running status and processing records. The emergence of BMP eliminates this restriction and significantly improves network monitoring efficiency.

Pre-configuration Tasks

Before configuring BMP, complete the following tasks:

- **Configure basic BGP functions.**
- Run the **active port-peering slot slot-id card card-id port port-list** command to activate peering gateway-specific licenses for the interfaces used to establish a BGP connection.

Procedure

- Configure BMP to report the BGP running status to a monitoring server.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bmp**

BMP is enabled, and the BMP view is displayed.
 - c. (Optional) Run **statistics-timer time**

The interval at which BMP sends BGP running statistics to a monitoring server is configured.

Configure the interval based on BGP stability requirements. In the case of high BGP stability requirements, configure a short interval. However, if the device frequently sends BGP running statistics, a large number of bandwidth resources will be consumed.
 - d. Run **bmp-session [vpn-instance vrf-name] ipv4-address [alias alias-name]**

An IPv4 session address is set for the TCP connection between the BMP device and the monitoring server.

alias alias-name specifies an alias for a BMP session. If the device needs to establish TCP connections with monitoring servers that have the same destination IP address but different destination port numbers, specify different values for the **alias alias-name** parameter to differentiate the connections.
 - e. Set the type of route whose statistics are to be sent by the device to the monitoring server.
 - Configure the device to send statistics about RIB-in routes (all received routes or only accepted routes) of BGP peers in a specified address family to the monitoring server.
 - 1) Run one of the following commands to enter the BMP-Monitor view:

- **monitor public:** displays the BMP-Monitor view and allows the BGP running status of all BGP peers in the public network address family to be monitored.
 - **monitor all-vpn-instance:** displays the BMP-Monitor view and allows the BGP running status of BGP peers in all VPN instance address families to be monitored.
 - **monitor peer:** displays the BMP-Monitor view and allows the BGP running status of a specified BGP peer in the public address family to be monitored.
 - **monitor vpn-instance:** displays the BMP-Monitor view and allows the BGP running status of all BGP peers in a specified VPN instance address family to be monitored.
 - **monitor vpn-instance peer:** displays the BMP-Monitor view and allows the BGP running status of a specified BGP peer in a specified VPN instance address family to be monitored.
- 2) Run **route-mode { ipv4-family unicast | ipv4-family labeled-unicast | ipv4-family vpnv4 } adj-rib-in { pre-policy | post-policy }**

The BMP device is configured to send statistics about RIB-in routes of BGP peers in a specified address family and the BGP running status of these peers to the monitoring server.

To configure the device to send statistics about all received routes to the monitoring server, specify **pre-policy** in the command. To configure the device to send statistics about only the routes that match the import policy (those delivered to the routing table) to the monitoring server, specify **post-policy** in the command.

 **NOTE**

If **pre-policy** is specified in the command, run the **keep-all-routes** command in the BGP view to save the routes carried in the BGP Update messages that are received from all BGP peers or peer groups after BGP connections are established, or run the **peer keep-all-routes** command to save the routes carried in the BGP Update messages that are received from a specified BGP peer or peer group after the BGP connection is established.

- Configure the device to send statistics about RIB-out routes of BGP peers in a specified address family to the monitoring server.
 - 1) Run one of the following commands to enter the BMP-Monitor view:
 - **monitor public:** displays the BMP-Monitor view and allows the BGP running status of all BGP peers in the public network address family to be monitored.
 - **monitor all-vpn-instance:** displays the BMP-Monitor view and allows the BGP running status of BGP peers in all VPN instance address families to be monitored.
 - **monitor peer:** displays the BMP-Monitor view and allows the BGP running status of a specified BGP peer in the public address family to be monitored.

- **monitor vpn-instance**: displays the BMP-Monitor view and allows the BGP running status of all BGP peers in a specified VPN instance address family to be monitored.
 - **monitor vpn-instance peer**: displays the BMP-Monitor view and allows the BGP running status of a specified BGP peer in a specified VPN instance address family to be monitored.
- 2) Run **route-mode { ipv4-family unicast | ipv4-family labeled-unicast | ipv4-family vpnv4 } adj-rib-out { pre-policy | post-policy }** The device is configured to send statistics about RIB-out routes of BGP peers in a specified address family and the BGP running status of these peers to the monitoring server.

If you want the monitoring server to monitor all the routes to be advertised, regardless of whether they match the export policy, specify **pre-policy** in the command. If you want the monitoring server to monitor only the routes that match the export policy, specify **post-policy** in the command.

- Configure the device to send statistics about Local-RIB routes of BGP peers in a specified address family to the monitoring server.
 - 1) Run one of the following commands to enter the BMP-Monitor view:
 - **monitor public**: displays the BMP-Monitor view and allows the BGP running status of all BGP peers in the public address family to be monitored.
 - **monitor vpn-instance**: displays the BMP-Monitor view and allows the BGP running status of all BGP peers in a specified VPN instance address family to be monitored.
 - 2) Run **route-mode { ipv4-family unicast | ipv4-family labeled-unicast | ipv4-family vpnv4 } local-rib [add-path | all] [path-marking]**

The BMP device is configured to send statistics about Local-RIB routes of BGP peers in a specified address family and the BGP running status of these peers to the monitoring server.

f. Run **quit**

Return to the BMP session view.

g. Run **tcp connect port port-number [password md5 cipher-password | keychain keychain-name]**

Parameters for the TCP connection to be established between the device and the monitoring server are configured.

 **NOTE**

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

h. (Optional) Run **ssl-policy name policy-name**

An SSL policy is configured for BMP.

 NOTE

Ensure that the specified SSL policy has been created using the **ssl policy *policy-name*** command in the system view.

- i. (Optional) Run **connect-interface { interface-type *interface-number* |
ipv4-source-address | interface-type *interface-number* ipv4-source-
address }**

The source interface for sending BMP messages is specified.

- j. Run **commit**

The configuration is committed.

 NOTE

If a configuration of a BMP session is changed and the new configuration needs to take effect immediately, run the **reset bmp session** command to reset the BMP session.

- Configure BMP to report the trace data of IPv4 public network unicast routes to the monitoring server.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bmp**

BMP is enabled, and the BMP view is displayed.
 - c. Run **bmp-session [vpn-instance *vrf-name*] *ipv4-address* [alias *alias-name*]**

An IPv4 BMP session address is specified for the TCP connection to be established between the device and monitoring server.
alias alias-name specifies an alias for a session. If the device needs to establish TCP connections with monitoring servers that have the same destination IP address but different destination port numbers, specify different values for the **alias alias-name** parameter to differentiate the connections.
 - d. Run **ipv4 unicast**

The BMP session IPv4 unicast view is created and displayed.
 - e. Run the **trace-prefix all** command to monitor the processing records of all IPv4 public network unicast routes, or run the **trace-prefix *ipv4-
address mask-length*** command to monitor the processing records of a specified IPv4 public network unicast route.
 - f. Run **commit**

The configuration is committed.
- Configure BMP to report the trace data of VPNV4 routes and the IPv4 VPN unicast routes (transformed from the VPNV4 routes) to the monitoring server.
 - a. Run **system-view**

The system view is displayed.
 - b. Run **bmp**

BMP is enabled, and the BMP view is displayed.

- c. Run **bmp-session [vpn-instance vrf-name] ipv4-address [alias alias-name]**

An IPv4 BMP session address is specified for the TCP connection to be established between the device and monitoring server.

alias alias-name specifies an alias for a session. If the device needs to establish TCP connections with monitoring servers that have the same destination IP address but different destination port numbers, specify different values for the **alias alias-name** parameter to differentiate the connections.

- d. Run **ipv4 vpn**

The BMP session IPv4 VPN view is created and displayed.

- e. Run the **trace-prefix route-distinguisher vrfRD all** command to configure the function of monitoring the processing records of all VPNV4 routes and IPv4 VPN unicast routes based on a specified RD. Alternatively, run the **trace-prefix route-distinguisher vrfRD ipv4-address mask-length** command to configure the function of monitoring the processing records of the VPNV4 route and IPv4 VPN unicast route based on a specified RD and route prefix.

- f. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bmp session [vpn-instance vrf-name] [ipv4-address [alias alias-name] verbose]** command to check BMP session configurations.
- Run the **display bgp bmp-monitor { all | { ipv4 | vpnv4 vpn-instance vpn-instance-name | vpnv4 } ipv4-address }** command to check information about BGP peers (either all peers or a specified one) monitored through BMP in all address families or in a specified address family.

1.1.9.2.45 Configuring BGP Route Dampening

BGP route dampening can be configured to suppress unstable routes.

Usage Scenario

Route instability is mainly reflected by route flapping. A route is considered to be flapping when it repeatedly appears and then disappears in the routing table. BGP is generally applied to complex networks where routes change frequently. Frequent route flapping consumes lots of bandwidth and CPU resources and even seriously affects network operations.

BGP route dampening can deal with frequent route flapping. It uses a penalty value to measure route stability. When a route flaps, it is assigned a penalty value. Later, each time the route flaps, its penalty value increases by a specific value. If

the penalty value of a route exceeds the pre-defined threshold, the route will not be advertised until the penalty value of the route reduces to the reuse threshold.

Pre-configuration Tasks

Before configuring BGP route dampening, complete the following task:

- **Configure basic BGP functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Configuring BGP route dampening parameters

- Configuring EBGP route dampening parameters
 - a. Run the **ipv4-family unicast** command to display the IPv4 unicast address family view.
 - b. Run the **dampening [half-life-reach reuse suppress ceiling | route-policy route-policy-name | route-filter route-filter-name] * [update-standard]** command to set the EBGP route dampening parameters.
- Configuring IBGP route dampening parameters
 - a. Run the **ipv4-family unicast** command to enter the IPv4 unicast address family view.
 - b. Run the **dampening ibgp [half-life-reach reuse suppress ceiling | route-policy route-policy-name | route-filter route-filter-name] * [update-standard]** command to set IBGP route dampening parameters.

When you configure BGP route dampening, the values of *reuse*, *suppress*, and *ceiling* should meet the relationship of *reuse*<*suppress*<*ceiling*.

If routes are differentiated based on policies and the **dampening** command is run to reference a route-policy, BGP can use different route dampening parameters to suppress different routes.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp routing-table flap-info [regular-expression as-regular-expression | as-path-filter { as-path-filter-number | as-path-filter-name } | network-address [{ mask | mask-length } [longer-match]]]** command to check route flapping statistics.

- Run the **display bgp routing-table time-range start-time end-time** command to check information about the routes that flap within a specified period.
- Run the **display bgp routing-table dampened** command to check dampened BGP routes.
- Run the **display bgp routing-table dampening parameter** command to check configured BGP route dampening parameters.

1.1.9.2.46 Configuring Suppression on BGP Peer Flapping

Suppression on BGP peer flapping allows a device to delay the establishment of a BGP peer relationship that flaps continuously.

Usage Scenario

BGP peer flapping occurs when BGP peer relationships are disconnected and then immediately re-established in a quick sequence that is repeated. Frequent BGP peer flapping is caused by various factors; for example, a link is unstable, or an interface that carries BGP services is unstable. After a BGP peer relationship is established, the local device and its BGP peer usually exchange all routes in their BGP routing tables with each other. If the BGP peer relationship is disconnected, the local device deletes all the routes learned from the BGP peer. Generally, a large number of BGP routes exist, and in this case, a large number of routes change and a large amount of data is processed when the BGP peer relationship is flapping. As a result, a high volume of resources are consumed, causing high CPU usage. To prevent this issue, a device supports suppression on BGP peer flapping. With this function enabled, the local device suppresses the establishment of the BGP peer relationship if it flaps continuously.

Pre-configuration Tasks

Before configuring suppression on BGP peer flapping, complete the following task:

- **Configure basic BGP functions.**

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **peer peerIpv4Addr oscillation-dampening**

BGP is enabled to suppress the establishment of a specified peer relationship that flaps continuously.

To immediately remove the suppression, you can run the **peer oscillation-dampening disable** command. Alternatively, you can run a reset command or another command that can cause the peer relationship to be disconnected and re-established.

Step 4 Run commit

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

Run the **display bgp peer verbose** command to check the status of suppression on BGP peer flapping and the remaining time to establish the BGP peer relationship.

1.1.9.2.47 Configuring Flapping Suppression Involved in BGP Next Hop Recursion

Flapping suppression involved in next-hop recursion prevents the system from frequently processing changes in routes that recurse to a frequently flapping next hop, thereby reducing system resource consumption and CPU usage.

Usage Scenario

If a large number of routes recurse to the same next hop that flaps frequently, the system will be busy processing changes of these routes, which consumes excessive system resources and leads to high CPU usage. To address this problem, configure flapping suppression involved in next-hop recursion.

By default, flapping suppression involved in next-hop recursion is enabled. After this function is enabled, BGP calculates the penalty value that starts from 0 by comparing the flapping interval with configured intervals if next hop flapping occurs. When the penalty value exceeds 10, BGP suppresses route recursion to the corresponding next hop. For example, if the intervals for increasing, retaining, and clearing the penalty value are T1, T2, and T3, respectively, BGP calculates the penalty value as follows:

- Increases the penalty value by 1 if the flapping interval is less than T1.
- Retains the penalty value if the flapping interval is greater than or equal to T1, but less than T2.
- Reduces the penalty value by 1 if the flapping interval is greater than or equal to T2, but less than T3.
- Clears the penalty value if the flapping interval is greater than or equal to T3.

Pre-configuration tasks

Before configuring flapping suppression involved in BGP next-hop recursion, [configure basic BGP functions](#).

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **undo nexthop recursive-lookup restrain disable**

Flapping suppression involved in next hop recursion is enabled.

If you do not want to slow down route recursion processing and high CPU usage due to route recursion change processing is not a concern, you can disable flapping suppression involved in next-hop recursion using the **nexthop recursive-lookup restrain disable** command.

Step 5 Run **quit**

Return to the BGP view.

Step 6 Run **nexthop recursive-lookup restrain suppress-interval add-count-time hold-interval hold-count-time clear-interval clear-count-time**

The intervals are configured for increasing, retaining, and clearing the penalty value for flapping suppression involved in next hop recursion.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp routing-table** command to check BGP public network route information.
- Run the **display bgp vpng4 routing-table** command to check BGP VPK4 and BGP VPN route information.

1.1.9.2.48 Configuring BGP-LS

BGP-LS provides a simple and efficient method of collecting topology information.

Usage Scenario

Without BGP-LS, the router uses an IGP (OSPF, OSPFv3, or IS-IS) to collect network topology information and report the topology information of each area to the controller separately. This method has the following disadvantages:

- The controller must have high computing capabilities and support the IGP and its algorithm.
- The controller cannot gain the complete inter-area topology information and therefore is unable to calculate optimal E2E paths.
- Different IGPs report topology information separately to the controller, which complicates the controller's analysis and processing.

With powerful route selection capabilities of BGP, BGP-LS has the following advantages:

- Reduces computing capability requirements and spares the necessity of IGPs on the controller.
- Facilitates route selection and calculation on the controller by using BGP to summarize process or AS topology information and report the complete information to the controller.
- Requires only one routing protocol (BGP) to report topology information to the controller.

BGP-LS needs to be deployed on the devices connected to the controller.

Pre-configuration Tasks

Before configuring BGP-LS, complete the following tasks:

- **Configure basic IPv4 IS-IS functions** or **basic OSPF functions** or **basic OSPFv3 functions**.

Procedure

1. Enable IGP topology advertisement to BGP. You can determine to enable IS-IS topology advertisement to BGP or OSPF topology advertisement to BGP based on network configurations.
 - Enable IS-IS topology advertisement to BGP.
 - i. Run **system-view**
The system view is displayed.
 - ii. Run **isis [process-id]**
An IS-IS process is configured.
 - iii. Run **bgp-ls enable [level-1 | level-2 | level-1-2] [exclude-prefix]**
IS-IS topology advertisement is enabled.
To enable IS-IS to advertise topology information of Level-1 areas and filter out the route prefixes leaked from Level-2 areas to Level-1 areas, run the **bgp-ls enable level-1 level-2-leaking-route-ignore** command.
 - iv. (Optional) Run **bgp-ls identifier identifier-value**
A BGP-LS identifier is configured for IS-IS.
 - v. (Optional) Run **bgp-ls report-exclude { metric-delay-average | metric-delay-variation | link-msd } ***
Filtering of IS-IS topology information to be advertised is configured.
 - vi. Run **commit**
The configuration is committed.
 - Enable OSPF topology advertisement.
 - i. Run **system-view**
The system view is displayed.
 - ii. Run **ospf [process-id | router-id router-id | vpn-instance vpn-instance-name] ***
An OSPF process is configured.

- iii. Run **bgp-ls enable**
OSPF topology advertisement to BGP is enabled.
 - iv. (Optional) Run **bgp-ls identifier identifier-value**
A BGP-LS identifier is configured for OSPF.
 - v. (Optional) Run **bgp-ls report-exclude { metric-delay-average | metric-delay-variation } ***
Filtering of the OSPF topology information to be advertised is configured.
 - vi. Run **commit**
The configuration is committed.
- Enable OSPFv3 topology advertisement.
- i. Run **system-view**
The system view is displayed.
 - ii. Run **ospfv3 [process-id] [vpn-instance vpn-instance-name]**
An OSPFv3 process is created.
 - iii. Run **bgp-ls enable**
OSPFv3 topology advertisement is enabled.
 - iv. (Optional) Run **bgp-ls identifier identifier-value**
A BGP-LS identifier is configured for OSPFv3.
 - v. (Optional) Run **bgp-ls report-exclude { metric-delay | metric-delay-average | metric-delay-variation } ***
Filtering of the OSPFv3 topology information to be advertised is configured.
- After BGP-LS is configured, the device collects OSPFv3 topology information and reports it to the controller. The collected information includes the maximum, minimum, and average delays and delay variation. If you want the device not to report some of the information, you can configure the topology advertisement filtering function.
- vi. Run **commit**
The configuration is committed.
2. Enable BGP-LS.
- a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
BGP is enabled, and the BGP view is displayed.
 - c. Run **peer { group-name | ipv4-address } as-number as-number**
The IP address and AS number of a BGP peer are specified.
 - d. Run **link-state-family unicast**
BGP-LS is enabled, and the BGP-LS address family view is displayed.
 - e. Run **peer { group-name | ipv4-address } enable**
BGP-LS route exchange with the specified peer or peer group is enabled.

- f. (Optional) Run **domain identifier domain-id**
A BGP-LS domain ID is configured.
A BGP-LS domain ID identifies a device with BGP-LS enabled. If no BGP-LS domain ID is configured, a BGP router ID is used as the BGP-LS domain ID by default. The same BGP-LS domain ID can be configured for multiple devices so that the controller calculates routes based on the combined topology information reported by the devices.
 - g. (Optional) Run **domain as domain-asNum**
A BGP-LS domain AS number is configured.
Two devices with different BGP AS numbers must have the same BGP-LS domain AS number configured using the **domain as** command so that the controller can obtain combined topology information about the two ASs for route calculation.
 - h. (Optional) Run **peer { group-name | ipv4-address } reflect-client**
An RR is configured, and a client is specified.
The router on which the **peer reflect-client** command is run functions as the RR, and the specified peer or peer group functions as a client.
-  NOTE
- If the clients of an RR are fully meshed, you can run the **undo reflect between-clients** command to disable route reflection among these clients through the RR to reduce bandwidth consumption.
- If a cluster has multiple RRs configured, you can run the **reflector cluster-id cluster-id** command to configure the same cluster ID for all the RRs. This command is applicable only to RRs.
- i. (Optional) Run **peer { group-name | ipv4-address } route-limit limit [percentage] [alert-only | idle-forever | idle-timeout minutes]**
The maximum number of BGP-LS routes that the local device can accept from a specified peer is set.
Generally, a BGP-LS routing table contains a large number of routes. If a large number of routes are received from peers, excessive system resources are consumed. To prevent this problem, run this command to set the maximum number of routes that a BGP device is allowed to accept from a peer. If **idle-forever** is configured, the peer relationship will be interrupted and will not be automatically re-established after the number of received routes exceeds the threshold. Therefore, configuring **idle-forever** is not recommended.
 - j. (Optional) Run **peer { group-name | ipv4-address } route-policy route-policy-name { import | export }**
A route-policy is specified for the BGP-LS routes to be received from or advertised to a specified BGP peer or peer group.
After a route-policy is created, you can run the **peer route-policy** command to use the route-policy to filter the BGP-LS routes to be received from or advertised to a specified BGP peer or peer group. The command configuration ensures that only desired routes are accepted or advertised, which helps manage routes and reduces the BGP-LS routing table size and system resource consumption.
 - k. (Optional) Run **peer { group-name | ipv4-address } route-update-interval interval**

An interval at which the device sends Update messages carrying the same route prefix to a specified peer or peer group is set.

When BGP-LS routes change, the router sends Update messages to notify its peers. If a route changes frequently, to prevent the router from sending Update messages for every change, run this command to set an interval at which the router sends Update messages carrying the same route prefix to a specified peer or peer group.

- l. (Optional) Run **peer { group-name | ipv4-address } allow-as-loop num**

The maximum number of AS number repetitions in the AS_Path attribute allowed by a peer or peer group is configured.

- m. Run **commit**

The configuration is committed.

Verifying the Configuration

After the configuration is complete, run the following commands to verify the configuration.

- Run the **display bgp link-state unicast peer** command to check information about BGP-LS peers and their status.
- Run the **display bgp link-state unicast routing-table** command to check BGP-LS route information.
- Run the **display bgp link-state unicast routing-table statistics** command to check BGP-LS route statistics.

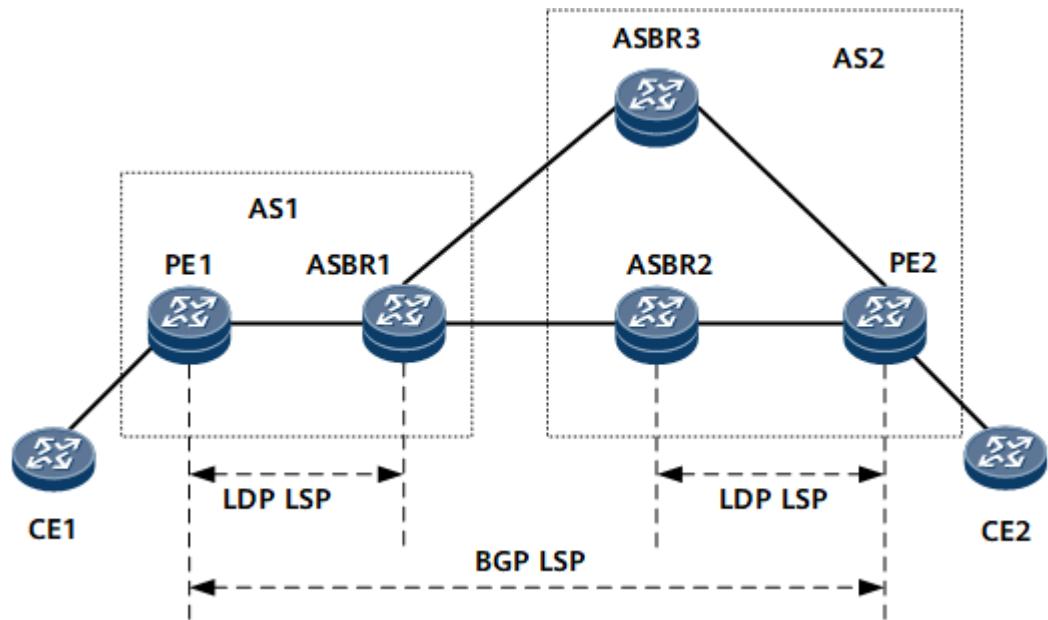
1.1.9.2.49 Configuring the Entropy Label Capability for a BGP LSP

Configuring the entropy label capability for a BGP LSP helps equalize and improve the performance of load balancing.

Usage Scenario

On the network shown in [Figure 1-402](#), an end-to-end BGP LSP is established between PE1 and PE2. In addition, an end-to-end VPNv4 IPv4 peer relationship is established between PE1 and PE2, an LDP LSP is established between PE1 and ASBR1, and an LDP LSP is established between ASBR2 and PE2. With the expansion of user networks, the load balancing technology is used to obtain higher bandwidth between nodes. However, load imbalance becomes increasingly severe on transit nodes. To improve load balancing performance, configure the entropy label capability for the BGP LSP on both PE1 and PE2.

Figure 1-402 Load balancing performed on transit nodes



Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 (Optional) Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 Run **peer { peerIpv4Addr | peerGroupName } advertise-entropy-label elc [padding paddingValue]**

The device is enabled to add the entropy label of the entropy label capability (ELC) type to BGP routes to be advertised to a specified peer or peer group.

Step 5 Run **peer { peerIpv4Addr | peerGroupName } entropy-label**

The device is enabled to add the entropy label information to traffic to be forwarded to the specified peer or peer group.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the following commands to verify the configuration.

Run the **display bgp routing-table** command to check the entropy label in BGP routes.

1.1.9.2.50 Configuring BGP RPD

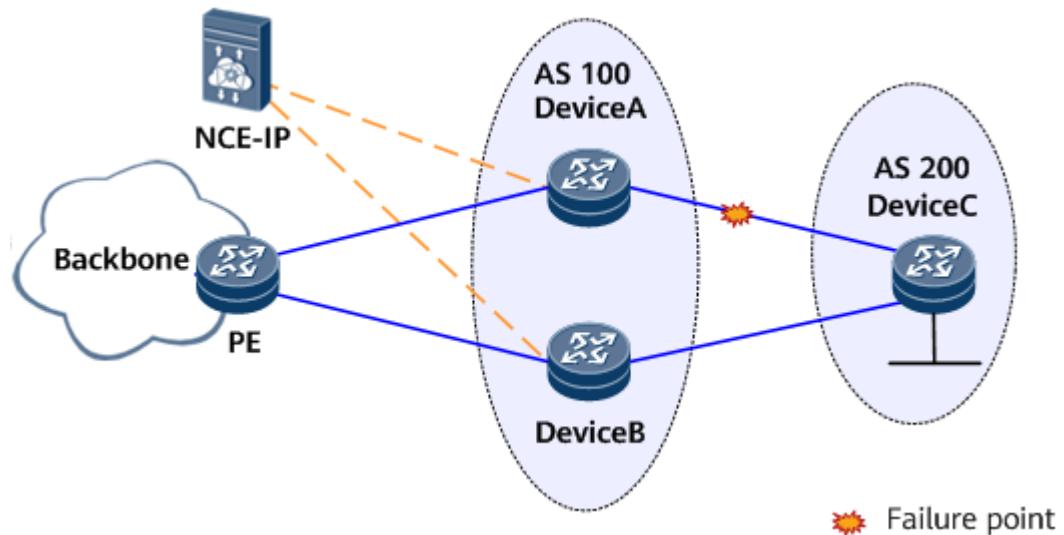
RPD provides a new method of distributing route-policies. It can work with the controller to efficiently and dynamically deploy route-policies.

Usage Scenario

In a MAN ingress or IGW scenario, uneven link resource usage or link faults may cause link congestion. To make full use of network bandwidth, you can deploy an inbound traffic optimization solution to adjust route priorities so that traffic is diverted to idle links. In such a scenario, the router functions as a forwarder, and RPD needs to be deployed on it.

In [Figure 1-403](#), an inbound traffic optimization solution is deployed so that traffic from AS 200 to the backbone network is monitored and scheduled in real time. If the link from Device C to Device A is congested, the traffic enters AS 100 through Device B and reaches the backbone network by path of the PE.

Figure 1-403 Typical networking of inbound traffic optimization



In this scenario, forwarders adjust route attributes based on the traffic optimization policies delivered by the controller. This ensures that routes are advertised based on the traffic optimization policies, which are configured on the controller based on traffic application. The following describes how to perform operations on the local device (forwarder).

Pre-configuration Tasks

Before configuring BGP RPD, complete the following tasks:

- [Configure basic BGP functions](#), and establish a connection between the device and the controller to ensure routing reachability between them.

- Run the **active port-peering slot slot-id card card-id port port-list** command to activate peering gateway-specific licenses for the interfaces used to establish a BGP connection.

Procedure

- Configure basic RPD functions.
 - Run **system-view**
The system view is displayed.
 - Run **bgp as-number**
The BGP view is displayed.
 - Run **rpd-family**
RPD is enabled, and the BGP RPD address family view is displayed.
 - Run **peer ipv4-address enable**
The device is enabled to exchange routing information with the specified peer.
 - Run **quit**
The BGP view is displayed.
 - Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - Run **peer ipv4-address rpd-policy export enable**
The RPD export route-policy function is enabled in the IPv4 unicast address family view.
 - Run **commit**
The configuration is committed.
- (Optional) Configure GR to prevent the traffic interruption caused by a protocol restart.
 - Run **system-view**
The system view is displayed.
 - Run **bgp as-number**
The BGP view is displayed.
 - Run **graceful-restart**
GR is enabled in the BGP view.
 - Run **rpd-family**
The BGP RPD address family view is displayed.
 - Run **peer ipv4-address graceful-restart static-timer restart-time**
GR is enabled in the BGP RPD address family, and a GR restart timer is set.
 - Run **commit**
The configuration is committed.

- (Optional) Configure router ID-based filtering on non-RRs in the RR scenario so that the non-RRs only accept the RPD routes that match the router ID of the local BGP process. If a large number of RPD routes are accepted, a large number of policy nodes are generated accordingly, which reduces the performance.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **rpd-family**
The BGP RPD address family view is displayed.
 - d. Run **router-id filter**
Router ID-based filtering is enabled.
- (Optional) Configure a delay for the protocol to apply an updated RPD route-policy after the controller notifies the local device of the RPD route update.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **ipv4-family unicast**
The IPv4 unicast address family view is displayed.
 - d. Run **rpd-policy change notify-delay delay-time**
A delay is configured for protocols to apply an updated RPD route-policy if the original policy changes.

----End

Verifying the Configuration

After configuring BGP RPD, verify the configuration.

- Run the **display bgp peer ipv4-address rpd export-policy** command to check RPD route-policies.
- Run the **display bgp rpd routing-table** command to check information about RPD routes.

1.1.9.2.51 Configuring the Capability of Creating MPLS Local IFNET Tunnels for BGP Peers

The MPLS local IFNET tunnels created by BGP peers can be used to carry BGP LSP traffic.

Usage Scenario

You can enable BGP peers to establish MPLS local IFNET tunnels to carry BGP LSP traffic.

Pre-configuration Tasks

Before configuring the capability of creating MPLS local IFNET tunnels for BGP peers, complete the following tasks:

- [Configure basic BGP functions.](#)

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **undo peer { peerGroupName | peerIpv4Addr } mpls-local-ifnet disable**

The capability of creating MPLS local IFNET tunnels is enabled for the specified IBGP peer.

NOTE

MPLS local IFNET tunnels can be established between IBGP peers only in the BGP-IPv4 unicast address family and BGP unicast labeled address family. To enable the capability of creating MPLS local IFNET tunnels to take effect in the BGP-IPv4 unicast address family, you also need to run the **peer label-route-capability** command in the BGP-IPv4 unicast address family view to enable the function of sending or receiving labeled routes.

Step 4 Run **quit**

The system view is displayed.

Step 5 (Optional) Configure MPLS local IFNET traffic statistics collection.

1. Run the **mpls** command to enable MPLS globally and enter the MPLS view.
2. Run the **quit** command to return to the system view.
3. Run the **mpls traffic-statistics** command to enable MPLS traffic statistics collection globally and enter the traffic statistics collection view.
4. Run the **bgp host [ip-prefix ip-prefix-name]** command to enable MPLS local IFNET traffic statistics collection. If you want statistics to be collected based on an IP prefix list (specified by **ip-prefix**), ensure that the IP prefix list has been created using the **ip ip-prefix** command.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, run the **display mpls lsp protocol bgp local-ifnet traffic-statistics outbound** command to query statistics about outgoing traffic in MPLS local IFNET tunnels.

1.1.9.2.52 Configuring the YANG Management Mode of BGP

In YANG management mode of BGP, BGP instance configurations can be delivered using the YANG model.

Usage Scenario

To manage BGP configurations using NETCONF YANG, you need to configure the YANG management mode of BGP on the device first. Alternatively, you can enable the leaf node (XPath: /bgp:bgp/bgp:global/bgp:yang-enable) globally through the YANG model file. This section describes the configuration on the device.

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

Procedure

Step 1 Run the **system-view** command to enter the system view.

Step 2 Run the **bgp yang-mode enable** command to configure the YANG management mode of the BGP VPN instance.

 NOTE

If the **bgp yang-mode enable** command is run, the configurations of BGP VPN instances and their peers and peer groups are changed. For example:

Example 1 (including part 1 and part 2):

Part 1: For example, before the **bgp yang-mode enable** command is run, the configurations on the device are as follows:

```
[~HUAWEI-bgp]display this
#
bgp 100
#
ipv4-family unicast
undo synchronization
#
ipv4-family vpn-instance abc
peer 3.3.3.4 as-number 100
#
ipv4-family vpn-instance vrf1
group a internal
peer 1.1.1.1 as-number 100
peer 1.1.1.1 group a
```

Part 2: After the **bgp yang-mode enable** command is run, the configurations on the device are changed as follows:

```
[~HUAWEI-bgp]display this
#
bgp 100
#
ipv4-family unicast
undo synchronization
#
vpn-instance abc //A BGP-VPN instance is added.
peer 3.3.3.4 as-number 100 //The VPN instance-level command configuration in the BGP-VPN instance IPv4 address family view is migrated to the new BGP-VPN instance view.
#
ipv4-family vpn-instance abc
peer 3.3.3.4 enable //The address-family level command is added to the BGP-VPN instance IPv4 address family view.
#
vpn-instance vrf1
group a internal //The VPN instance-level command configuration in the BGP-VPN instance IPv4 address family view is migrated to the new BGP-VPN instance view.
peer 1.1.1.1 as-number 100 //The VPN instance-level command configuration in the BGP-VPN instance IPv4 address family view is migrated to the new BGP-VPN instance view.
peer 1.1.1.1 group a //The VPN instance-level command configuration in the BGP-VPN instance IPv4 address family view is migrated to the new BGP-VPN instance view.
#
ipv4-family vpn-instance vrf1
peer a enable //The address-family level command is added to the BGP-VPN instance IPv4 address family view.
peer 1.1.1.1 enable //The address-family level command is added to the BGP-VPN instance IPv4 address family view.
peer 1.1.1.1 group a enable //The address-family level command is added to the BGP-VPN instance IPv4 address family view.
```

- After the **bgp yang-mode enable** command is run, the configurations shown in part 1 of example 1 cannot be performed.
- After the **bgp yang-mode enable** command is run, if a BGP-VPN instance is deleted, all the configurations in the instance are deleted accordingly.

Example 2: Example 2 is based on example 1. In this example, peer groups of the same name and in the same BGP VPN instance are changed.

Before the **bgp yang-mode enable** command is run, the configurations on a device are as follows:

```
[~HUAWEI-bgp]display this
#
```

```
bgp 100
#
ipv4-family unicast
undo synchronization
#
ipv4-family vpn-instance vrf1
group ML internal
peer ML password simple 11
#
ipv6-family vpn-instance vrf1
group ML internal
peer ML connect-interface LoopBack1
```

After the **bgp yang-mode enable** command is run, the configurations on the device are changed as follows:

```
[~HUAWEI-bgp]display this
#
bgp 100
#
ipv4-family unicast
undo synchronization
#
vpn-instance vrf1
group ML internal
peer ML password simple 11
group ML2019531105624 internal //In this example, the name of the peer group changes from ML to ML2019531105624.
peer ML2019531105624 connect-interface LoopBack1 //In this example, the name of the peer group changes from ML to ML2019531105624.
#
ipv4-family vpn-instance vrf1
peer ML enable
#
ipv6-family vpn-instance vrf1
peer ML2019531105624 enable
```

Step 3 Run the **commit** command to commit the configuration.

----End

1.1.9.2.53 Improving BGP Security

To improve BGP network security, you can configure BGP authentication, RPKI, and GTSM on the BGP network.

Usage Scenario

You can configure the following functions to improve BGP network security:

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

- MD5 authentication

BGP uses TCP as the transport protocol and considers a packet valid if the source address, destination address, source port, destination port, and TCP sequence number of the packet are correct. However, most parameters in a packet are easily accessible to attackers. To protect BGP against attacks, configure MD5 authentication for TCP connections established between BGP peers.

To prevent the MD5 password set on a BGP peer from being decrypted, update the MD5 password periodically.

 NOTE

The MD5 algorithm is not recommended if high security is required.

- Keychain authentication

A keychain consists of multiple authentication keys, each of which contains an ID and a password. Each key has a lifecycle, and keys are dynamically selected based on the life cycle of each key. After a keychain with the same rules is configured on the two ends of a BGP connection, the keychains can dynamically select authentication keys to enhance BGP attack defense.

- BGP GTSM

The GTSM mechanism protects the router by checking whether the TTL value in an IP packet header is within a pre-defined range, which enhances the system security.

- BGP RPKI

Resource Public Key Infrastructure (RPKI) improves BGP security by validating the origin ASs of BGP routes.

- SSL/TLS authentication

Secure Sockets Layer (SSL) is a security protocol that protects data privacy on the Internet. Transport Layer Security (TLS) is a successor of SSL. TLS protects data integrity and privacy by preventing attackers from eavesdropping the data exchanged between a client and server. To ensure data transmission security on a network, SSL/TLS authentication can be enabled for BGP message encryption.

- TCP-AO authentication

The TCP authentication option (TCP-AO) is used to authenticate received and to-be sent packets during TCP session establishment and data exchange. It supports packet integrity check to prevent TCP replay attacks. TCP-AO authentication improves the security of the TCP connection between BGP peers and is applicable to the network that requires high security.

 NOTE

GTSM supports only unicast addresses. Therefore, configure GTSM on all the routers configured with routing protocols.

Pre-configuration Tasks

Before configuring BGP security, complete the following tasks:

- [Configure basic BGP functions.](#)
- Configure a keychain.

Configuring MD5 Authentication

In MD5 authentication, a Message Digest 5 (MD5) authentication password is set for a TCP connection, and the MD5 authentication is performed by TCP. If authentication fails, no TCP connection will be established.

Context

NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The encryption algorithm used for MD5 authentication is insecure and poses security risks. As such, you are advised to use a more secure encryption algorithm.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **peer { ipv4-address | group-name } password { cipher cipher-password | simple simple-password }**

An MD5 authentication password is set.

An MD5 authentication password can be set in either of the following modes:

- **cipher cipher-password** indicates that a password is set using a ciphertext string.
- **simple simple-password** indicates that a password is set using a plaintext string.

NOTE

- The new password is at least eight characters long and contains at least two of upper-case letters, lower-case letters, digits, and special characters.
- When configuring an authentication password, select the ciphertext mode because the password is saved in configuration files in simple text if you select the simple text mode, which has a high risk. To ensure device security, change the password periodically. If this command is run in the BGP view, the configuration also takes effect in an extended BGP address family view because they use the same TCP connection. BGP MD5 authentication and BGP keychain authentication are mutually exclusive.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After completing the configuration, verify it.

Run the **display bgp peer [ipv4-address] verbose** command to view the authentication information about BGP peers.

Configuring Keychain Authentication

After a keychain with the same rules is configured on the two ends of a BGP connection, the keychain can dynamically select the authentication keys to enhance BGP attack defense.

Procedure

- Configuring Keychain Authentication.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **bgp as-number**
The BGP view is displayed.
 - c. Run **peer { ipv4-address | group-name } keychain keychain-name**
Keychain authentication is configured.

To ensure the setup of a TCP connection and BGP exchange between on both ends of a BGP connection, configure keychain authentication specified for TCP-based applications and the same password and encryption algorithms on both ends.

keychain-name specified in this command must exist; otherwise, the TCP connection cannot be established. For keychain configuration details, see the "Keychain Configuration" chapter in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Configuration Guide - Security*.

NOTE

- When this command is used in the BGP view, it is also applicable to the extended address family view because they use the same TCP connection.
- BGP MD5 authentication and BGP keychain authentication are mutually exclusive.

- d. Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the following command to check the previous configuration.

Run the **display bgp peer [ipv4-address] verbose** command to view the authentication information about BGP peers.

Configuring TCP-AO Authentication

This section describes how to configure BGP TCP Authentication Option (TCP-AO) authentication to check the integrity of packets and prevent TCP replay attacks.

Context

The TCP-AO is used to authenticate received and to-be sent packets during TCP session establishment and data exchange. It supports packet integrity check to

prevent TCP replay attacks. After creating a TCP-AO, run the **peer tcp-ao policy** command in the BGP view and specify the peer that needs to reference the TCP-AO and the TCP-AO name. This enables the BGP session to be encrypted. Such configuration is applicable to networks that require high security. Different peers can reference the same TCP-AO.

Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** Run the **tcp ao *tcpaoName*** command to create a TCP-AO and enter the TCP-AO policy view.
- Step 3** Run the **binding keychain *kcName*** command to bind the TCP-AO to a keychain.
- Step 4** Run the **key-id *keyId*** command to create a key ID for the TCP-AO and enter the TCP-AO key ID view.
- Step 5** Run the **send-id *snId* receive-id *rcvId*** command to configure send-id and receive-id for the Key ID.
- Step 6** Run the **quit** command to return to the upper-level view.
- Step 7** Run the **quit** command to return to the system view.
- Step 8** Run the **bgp *as-number*** command to enter the BGP view.
- Step 9** Run the **peer *ipv4-address* **as-number** *as-number*** command to specify the IP address of a peer and the number of the AS where the peer resides.
- Step 10** Run the **peer *ipv4-address* **tcp-ao policy** *tcp-ao-name*** command to configure TCP-AO authentication for the TCP connection to be set up between BGP peers.

The value of the *tcp-ao-name* parameter must be set to the TCP-AO created in step 2.



For the same peer, the authentication modes TCP-AO, MD5, and keychain are mutually exclusive.

- Step 11** Run the **commit** command to commit the configuration.

----End

Configuring BGP GTSM

BGP GTSM must be configured on both peers.

Usage Scenario

GTSM prevents attacks through TTL detection. An attacker simulates real BGP packets and sends the packets in a large quantity to the router. After receiving the packets, an interface board of the router directly sends the packets to the BGP

module of the control plane if the interface board finds that the packets are sent by the local router, without checking the validity of the packets. The control plane of the router needs to process the "legal" packets. As a result, the system becomes abnormally busy and the CPU usage is high.

GTSM protects the router by checking whether the TTL value in an IP packet header is within a pre-defined range to enhance the system security.

 NOTE

- GTSM supports only unicast addresses; therefore, GTSM must be configured on all the routers configured with routing protocols.

Pre-configuration Tasks

Before configuring the BGP GTSM, complete the following task:

- [Configuring Basic BGP Functions](#)

Perform the following steps on both BGP peers:

Procedure

Step 1 Configure the basic BGP GTSM functions.

1. Run **system-view**

The system view is displayed.

2. Run **bgp as-number**

The BGP view is displayed.

3. Run **peer { group-name | ipv4-address } valid-ttl-hops [hops]**

The BGP GTSM is configured.

The valid TTL range of detected packets is [255 - *hops* + 1, 255]. For example, for an EBGP direct route, the number of hops is 1, that is, the valid TTL value is 255

 NOTE

- When being configured in the BGP view, GTSM is also applicable to MP-BGP VPNv4 extensions because they use the same TCP connection.
- The GTSM and EBGP-MAX-HOP functions both affect the TTL values of sent BGP messages and they conflict with each other. Thus, for a peer or a peer group, you can use only either of them.

A BGP router that is enabled with GTSM checks the TTL values in all BGP packets. As required by the actual networking, packets whose TTL values are not within the specified range are discarded. If GTSM is not configured on a BGP router, the received BGP packets are forwarded if the BGP peer configuration is matched. Otherwise, the received BGP packets are discarded. This prevents bogus BGP packets from consuming CPU resources.

4. Run **commit**

The configuration is committed.

Step 2 Set the default action for packets that do not match the GTSM policy.

GTSM only checks the TTL values of packets that match the GTSM policy. Packets that do not match the GTSM policy can be allowed or dropped. If "drop" is set as the default GTSM action for packets, you need to configure TTL values for all the packets sent from valid peers in the GTSM policy. If TTL values are not configured for the packets sent from a peer, the device will discard the packets sent from the peer and cannot establish a connection to the peer. Therefore, GTSM enhances security but reduces the ease of use.

You can enable the log function to record packet drop for troubleshooting.

Perform the following configurations on the GTSM-enabled router:

1. Run **system-view**

The system view is displayed.

2. Run **gtsm default-action { drop | pass }**

The default action for packets that do not match the GTSM policy is configured.

 **NOTE**

If the default action is configured but no GTSM policy is configured, GTSM does not take effect.

This command is supported only on the Admin-VS and cannot be configured in other VSs. This command takes effect on all VSs.

3. Run **commit**

The configuration is committed.

----End

Checking the Configurations

Run the following command to check the previous configurations.

- Run the **display gtsm statistics { slot-id | all }** command to check the statistics about GTSM.

 **NOTE**

In VS mode, this command is supported only by the admin VS.

Configuring ROA

Resource Public Key Infrastructure (RPKI) can validate the origin of BGP routes, ensuring BGP security. Using RPKI, Route Origin Authorization (ROA) can validate and filter routes.

Usage Scenario

To solve the problem of BGP route hijacking, the industry proposes the RPKI solution that validates the origin ASs of BGP routes. Distributed RPKI servers are used to collect information such as the origin AS numbers, route prefixes, and masks of BGP routes initiated by each ISP. After a device sets up a connection with

an RPKI cache server, the device saves a copy of ROA data locally. If no RPKI server is available, static ROA data can be configured on a device. Inbound ROA validation, which applies to the BGP routes received from peers, can be configured to control route selection. In addition, outbound ROA validation, which applies to the BGP routes to be advertised to peers, can be configured to control route advertisement. ROA validation ensures that hosts in an AS can securely access external services.

Pre-configuration Tasks

Before configuring ROA, complete the following tasks:

- **Configure basic BGP functions.**
- Run the **active port-peering slot slot-id card card-id port port-list** command to activate an interface-specific peering scenario license.

Procedure

Step 1 Select one of the following configurations based on the usage scenario:

- If the local device needs to obtain data from the ROA database through a connection to be established with an RPKI server, perform the following operations:
 - a. Run **system-view**
The system view is displayed.
 - b. Run **rpkı**
RPKI is started, and the RPKI view is displayed.
 - c. Run **session ipv4-address**
An address of the RPKI server is specified for a TCP connection to be set up between the device and the RPKI server.
 - d. Run **tcp port port-number [password md5 cipher-password | keychain keychain-name]**
Parameters are configured for the TCP connection to be set up between the device and the RPKI server.

NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

The password must be at least eight characters long and contain at least two of the following character types: uppercase letters, lowercase letters, digits, and special characters (excluding question marks and spaces).

For security purposes, you are advised to configure a ciphertext password, and change it periodically.

- e. (Optional) Run **timer { aging aging-time | refresh refresh-time }**

Timers are configured for the RPKI session.

The *aging-time* parameter specifies a value of the validation data age timer, and the *refresh-time* parameter specifies a value of the session

update timer. You can configure the timers based on actual requirements on BGP security. Small values are recommended if high BGP security is required. However, frequent data updates consume much network bandwidth.

- f. (Optional) Run **rpk-limit** *limit* [**alert-only** | **idle-forever** | **idle-timeout times**]

The maximum number of ROA entries that the device is allowed to accept in a session is configured.

In most cases, a large number of ROA entries exist on an RPKI server. If the device receives a large number of ROA entries from the RPKI server, excessive system resources will be consumed. To prevent this problem, run the **rpk-limit** command to configure the maximum number of ROA entries that the BGP device is allowed to accept in a session.

- g. (Optional) Run **connect-interface** { *interface-name* | *ipv4-source-address* | *interface-type interface-number* | *interface-type ipv4-source-address* | *interface-type interface-number ipv4-source-address* }

The source interface for sending RPKI packets is specified.

- h. (Optional) Run **ssl-policy** *policy-name*

An SSL policy to be bound to the TCP connection between the device and RPKI server is configured.

- i. Run **quit**

The RPKI view is displayed.

- j. Run **quit**

The system view is displayed.

- k. Run **commit**

The configuration is committed.

NOTE

After RPKI session configurations are changed, run the **reset rpk-session** command to reset the involved RPKI session for the new configurations to take effect.

- If a static ROA database needs to be configured on the local device, perform the following operations:
 - a. Run **system-view**
The system view is displayed.
 - b. Run **rpk**
RPKI is started, and the RPKI view is displayed.
 - c. Run **origin-validation**
A static ROA database is created, and the RPKI origin-validation view is displayed.
 - d. Run **static record** *ipv4-address mask-length max-length max-mask-length origin-as as-number*
A record is configured for the static ROA database.
 - e. Run **quit**
The RPKI view is displayed.

- f. Run **quit**
The system view is displayed.
- g. Run **commit**
The configuration is committed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Configure inbound or outbound ROA validation as required.

- To configure inbound ROA validation (validation results not affecting route acceptance) for the routes received from an EBGP peer, perform the following operations:
 - a. Run **prefix origin-validation enable**
Origin AS validation of RPKI is enabled.
After origin AS validation is enabled, the device matches the origin AS of each received route against the origin AS data in the database and provides the validation result, which can be Valid, Not Found, or Invalid.
 - b. (Optional) Run **bestroute origin-as-validation [allow-invalid]**
The device is configured to apply origin AS validation results of RPKI to BGP route selection.
BGP selects routes in descending order of Valid, Not Found, and Invalid after origin AS validation results are applied to route selection. If **allow-invalid** is not specified in the command, the BGP routes with the validation result being Invalid do not participate in route selection.
 - c. (Optional) Run **peer { ipv4-address | group-name } advertise-ext-community**
The device is configured to advertise extended community attributes to the specified peer.
 - d. (Optional) Run **peer { ipv4-address | group-name } advertise origin-as-validation**
The device is enabled to advertise origin AS validation results of RPKI to the specified BGP peer or peer group.

 **NOTE**

Origin AS validation results of RPKI can be advertised only to IBGP peers.

- To configure outbound ROA validation for the routes to be advertised to an EBGP peer to control route advertisement, perform the following operations:
Run peer { peerIpv4Addr | peerGroupName } origin-validation export [include-not-found [external]]
The local device is configured to perform outbound ROA validation on the routes to be advertised to the specified EBGP peer.
After the local device is configured to perform outbound ROA validation on the routes to be advertised to a specified EBGP peer, the device matches the origin ASs of the routes against those of the matched routes recorded in the database. The validation result can be Valid, Not Found, or Invalid. By default, only the routes whose validation result is Valid are advertised. To configure the device to advertise the routes with the validation result being Valid or Not

Found, specify the **include-not-found** keyword in the preceding command. To configure the device to advertise the routes with the validation result being Valid or Not Found (if the routes with the result being Not Found were received from another AS), specify the **include-not-found external** keyword in the preceding command.

Step 4 Run commit

The configuration is committed.

----End

Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display rpki session *ipv4-address* verbose** command to check RPKI session configurations.
- Run the **display rpki table** command to check ROA information.

Configuring ASPA

RPKIV2-based ASPA validation validates the AS_Path attribute of routes, ensuring BGP security.

Usage Scenario

RPKIV0-based ROA validation can detect unexpected path leaks, but it relies on the origin AS in the BGP attribute AS_Path, which may be manipulated by attackers. As an inter-AS routing security mechanism, RPKIV0 provides only origin validation but not path validation.

ASPA can automatically detect invalid AS_Paths in routes received from peers based on the customer-to-provider shared signature database constructed through RPKIV2.

Pre-configuration Tasks

Before configuring ASPA, complete the following task:

- [Configure basic BGP functions.](#)

Procedure

Step 1 Select one of the following configurations based on the usage scenario:

- If the local device needs to obtain data in the ASPA database through a connection to be established with an RPKI server, perform the following operations:
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **rpki** command to start RPKI and enter the RPKI view.
 - c. Run the **session *ipv4-address*** command to configure session information for the TCP connection between the local device and the RPKI server.

- d. Run the **tcp port** *port-number* [**password md5** *cipher-password* | | **keychain** *keychain-name*] command to configure parameters for the TCP connection between the local device and the RPKI server.

 NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

The password must be at least eight characters long and contain at least two of the following character types: uppercase letters, lowercase letters, digits, and special characters (excluding question marks and spaces).

For security purposes, you are advised to configure a ciphertext password, and change it periodically.

- e. Run the **version** *version-num* command to configure an RPKI version. RPKIv2 allows a device to receive ASPA data on the condition that both the device and the RPKI server support RPKIv2.
- f. (Optional) Run the **timer { aging** *aging-time* | **refresh** *refresh-time* } command to configure timers for the RPKI session.

The *aging-time* parameter specifies a value of the validation data aging timer, and the *refresh-time* parameter specifies a value of the session update timer. You can configure the timers based on actual requirements on BGP security. Small values are recommended if high BGP security is required. However, frequent data updates consume much network bandwidth.

- g. (Optional) Run the **aspalimit** *limit* [*percentage*] [**alert-only** | **idle-forever** | **idle-timeout** *times*] command to configure the maximum number of ASPA pairs that the device is allowed to accept in the session.

In most cases, a large number of ASPA pairs exist on a server. If a BGP device receives a large number of ASPA pairs from the server, excessive system resources will be consumed. To prevent this problem, run the **aspalimit** command to configure the maximum number of ASPA pairs that the device is allowed to accept in the session.

- h. (Optional) Run the **connect-interface** { *interface-name* | *ipv4-source-address* | *interface-type interface-number* | *interface-type ipv4-source-address* | *interface-type interface-number ipv4-source-address* } command to specify the source interface for sending RPKI messages.
- i. (Optional) Run the **ssl-policy** *policy-name* command to configure an SSL policy to be bound to the TCP connection between the device and the RPKI server.
- j. Run the **quit** command to enter the RPKI view.
- k. Run the **quit** command to enter the system view.
- l. Run the **commit** command to commit the configuration.

 NOTE

If configurations of an RPKI session are changed and you want its new configurations to take effect immediately, run the **reset rpkisession** command to reset the RPKI session.

- If a static ASPA database needs to be configured on the local device, perform the following operations:
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **rplki** command to start RPKI and enter the RPKI view.
 - c. Run the **aspa-validation** command to create a static ASPA database and enter the RPKI ASPA-validation view.
 - d. Run the **static record customer-as provider as-number { ipv4 | ipv6 }** command to configure a static ASPA database.
 - e. Run the **quit** command to enter the RPKI view.
 - f. Run the **quit** command to enter the system view.
 - g. Run the **commit** command to commit the configuration.

Step 2 Run the **bgp as-number** command to enter the BGP view.

Step 3 Configure inbound ASPA validation as required. To configure inbound ASPA validation (validation results not affecting route acceptance) for the routes received from an EBGP peer, perform the following operations:

1. Run the **peer { peerIpv4Addr | peerGroupName } role { provider | rs | rs-client | customer | lateral-peer | sibling }** command to configure a role for a BGP peer.
2. Run the **aspa-validation enable** command to enable RPKI-based ASPA validation. After ASPA validation is enabled, the device compares the AS_Path of a route with the matching ASPA pair recorded in the database and provides the validation results: Valid, NotFound, or Invalid.
3. (Optional) Run the **bestroute aspa-validation [allow-invalid]** command to configure the device to apply ASPA validation results of RPKI to BGP route selection. BGP selects routes in descending order of Valid, Not Found, and Invalid after ASPA validation results are applied to route selection. If **allow-invalid** is not specified in the command, the BGP routes with the validation result being Invalid do not participate in route selection.

Step 4 Run the **commit** command to commit the configuration.

----End

Verifying the Configuration

After the configuration is complete, verify it.

Run the **display rplki aspa table** command to check ASPA-related data.

Configuring Regional Validation

Resource Public Key Infrastructure (RPKI) regional validation or regional confederation validation ensures BGP security by validating route advertisers.

Usage Scenario

Regional validation: Users can manually combine multiple trusted ASs into a region and combine multiple regions into a regional confederation. Regional validation controls route selection results by checking whether the routes received from EBGP peers in an external region belong to the local region. This prevents

intra-region routes from being hijacked by attackers outside the local region, and ensures that hosts in the local region can securely access internal services.

Regional validation applies to the following typical scenarios: regional validation scenario and regional confederation validation scenario.

Pre-configuration Tasks

Before configuring regional validation, complete the following tasks:

- **Configure basic BGP functions.**
- Run the **active port-peering slot slot-id card card-id port port-list** command to activate an interface-specific peering scenario license.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run rpkı

RPKI is started, and the RPKI view is displayed.

Step 3 Run region-validation

Regional validation is enabled, and the region-validation view is displayed.

Step 4 You can configure regions or regional confederations as required.

- Create a region.
 - a. Run **region region-id**
A region is created.
 - b. Run **description description-text**
A description is configured for the region.
 - c. Run **as-number { asn } &<1-100>**
An AS number list is configured so that the AS numbers in it can be added to the region.
 - d. Run **quit**
The RPKI view is displayed.
 - e. Run **quit**
The system view is displayed.
- Create a regional confederation.
 - a. Run **region region-id**
A region is created.
 - b. Run **quit**
Exit the RPKI region-validation-region view.
 - c. Run **region-confederation region-confederation-id**
A regional confederation is created.
 - d. Run **description description-text**

- A description is configured for the regional confederation.
- e. Run **region { region-id } &<1-100>**
A region ID list is configured in the regional confederation so that regions in the list are added to the regional confederation.
 - f. Run **quit**
The RPKI view is displayed.
 - g. Run **quit**
The system view is displayed.

Step 5 Run **bgp as-number**

The BGP view is displayed.

Step 6 Run the **ipv4-family unicast** command to display the IPv4 unicast address family view.

Step 7 Enable the region or regional confederation function as required.

- Run **region-validation**
BGP regional validation is enabled.
- Run **region-validation confed-check strict**
Strict BGP regional validation is enabled.

Step 8 Run **bestroute region-validation [allow-invalid]**

The device is configured to apply the BGP regional validation results of RPKI to BGP route selection.

If regional validation succeeds, the route is valid and can participate in route selection. If regional validation fails, the route is invalid and cannot participate in route selection. To allow the routes that fail regional validation to be valid and participate in route selection, configure the **allow-invalid** parameter in the command. The priority of such routes is reduced during route selection.

Step 9 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the **display rpki session ipv4-address verbose** command to check RPKI session configurations.

Enabling SSL/TLS Authentication for BGP

To ensure data transmission security on a network, SSL/TLS authentication can be enabled for BGP message encryption.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ssl policy *policy-name***

An SSL policy is created, and the SSL policy view is displayed.

Step 3 Run **quit**

Return to the system view.

Step 4 Run **bgp *as-number***

The BGP view is displayed.

Step 5 To configure SSL/TLS authentication for BGP, perform the following steps (no sequential order) on the client and server (the priority of the configuration on a peer is higher than that of the configuration on the peer group):

- To configure a peer or peer group as an SSL client or server, run the **peer { group-name | ipv4-address } ssl-policy role { client | server }** command.
- To apply the SSL policy to the SSL client or server, run the **peer { group-name | ipv4-address } ssl-policyname *ssl-policy-name*** command.
- To enable SSL/TLS authentication, run the **peer { group-name | ipv4-address } ssl-server certificate** command.



This operation can be performed only on the server.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After enabling SSL/TLS authentication for BGP, verify the configuration.

Run the **display bgp peer [*ipv4-address*] verbose** command to check the authentication information of BGP peers.

1.1.9.2.54 Configuring BGP Extensions

Configuring BGP extensions enables BGP to provide routing information for multiple routing protocols.

Usage Scenario

The section does not describe the commands that are associated with specific applications and used in the MP-BGP address family view in detail. For details, see the MP-BGP chapter.

Pre-configuration Tasks

Before configuring BGP extensions, complete the following task:

- [Configure basic BGP functions.](#)

1.1.9.2.55 Configuring BGP Multi-Instance

You can configure BGP multi-instance to achieve separate route management and maintenance.

Usage Scenario

By default, all BGP routes are stored in the BGP base instance, and separate route management and maintenance are impossible. To address this problem, BGP multi-instance is introduced. A device can simultaneously run the BGP base instance and BGP multi-instance, which are independent of each other and can have either the same AS number or different AS numbers. You can deploy different address families for the BGP base instance and BGP multi-instance as required to implement separate route management and maintenance.

Procedure

- Step 1** Run the **system-view** command to enter the system view.
- Step 2** Run the **ip vpn-instance *vpn-instance-name*** command to create a VPN instance and enter its view.
- Step 3** Run the **ipv4-family** command to enable the VPN instance IPv4 address family and enter its view.
- Step 4** Run the **route-distinguisher *route-distinguisher*** command to configure an RD for the VPN instance IPv4 address family.
- Step 5** Run the **vpn-target *vpn-target* &<1-8> [both | export-extcommunity | import-extcommunity]** command to configure VPN targets for the VPN instance IPv4 address family.
- Step 6** Run the **quit** command to enter the VPN instance view.
- Step 7** Run the **quit** command to enter the system view.
- Step 8** Run the **bgp *as-number* instance *instance-name*** command to enter the BGP multi-instance view.
- Step 9** (Optional) Run the **ipv4-family vpn-instance *vpn-instance-name*** command to enter the BGP multi-instance VPN instance IPv4 address family view.
- Step 10** Run the **peer *ipv4-address* as-number *as-number*** command to specify the IP address of a peer and the number of the AS where the peer resides.
The IP address of the peer can be one of the following types:
 - IP address of the peer's interface that is directly connected to the local device
 - IP address of a sub-interface on the peer's interface that is directly connected to the local device
 - IP address of a reachable loopback interface on the peer
- Step 11** Run the **commit** command to commit the configuration.

----End

1.1.9.2.56 Maintaining BGP

Maintaining BGP involves resetting BGP connections and clearing BGP statistics.

Resetting BGP Connections

Resetting a BGP connection will interrupt peer relationships.

Context

NOTICE

The BGP peer relationship between routers is interrupted after you reset BGP connections with the **reset bgp** command. Exercise caution when resetting BGP connections.

When the BGP routing policy on the router that does not support the route-refresh capability changes, you need to reset BGP connections so that the change can take effect. To reset BGP connections, run the following reset commands in the user view:

Procedure

- To reset all BGP connections, run the **reset bgp all** command.
- To reset BGP connections with a specified AS, run the **reset bgp as-number** command.
- To reset BGP connections with a specified peer, run the **reset bgp ipv4-address** command.
- To reset all EBGP connections, run the **reset bgp external** command.
- To reset BGP connections with a specified peer group, run the **reset bgp group group-name** command.
- To reset all IBGP connections, run the **reset bgp internal** command in the user view.
- To reset the BGP connection with a specified slow peer, run the **reset bgp ipv4 ipv4-address slow-peer** command in the user view.

----End

Clearing BGP Statistics

This section describes how to clear BGP statistics about flapping routes and dampened routes.

Procedure

- To clear statistics about flapping routes, run the **reset bgp flap-info [regexp as-path-regexp | as-path-filter { as-path-filter-number | as-path-filter-name } | ipv4-address [mask | mask-length]]** command in the user view.

NOTICE

BGP statistics cannot be restored after being cleared. Therefore, exercise caution when running the command.

- To clear statistics about flapping routes of a specified peer, run the **reset bgp *ipv4-address* flap-info** command in the user view.
- To clear statistics about dampened routes and release dampened routes, run the **reset bgp dampening [*ipv4-address* [*mask* | *mask-length*]]** command in the user view.

----End

Clearing the SIDs That Are in Delayed Deletion State in BGP

This section describes how to clear BGP statistics about flapping routes and dampened routes. SIDs in the delayed deletion state consume SID resources. If SID resources are insufficient, other services may fail to apply for SIDs. In this case, you can clear the SIDs in the delayed deletion state from BGP.

Procedure

- To immediately clear all SIDs in the delayed deletion state from BGP, run the **reset bgp slow-delete sid all** command in the user view.
- To immediately clear the SIDs in the delayed deletion state from the BGP IPv4 unicast address family, run the **reset bgp slow-delete sid ipv4** command in the user view.
- To immediately clear the SIDs in the delayed deletion state from the BGP-VPN instance IPv4 address family, run the **reset bgp slow-delete sid ipv4 vpn** command in the user view.
- To immediately clear the End.DT46 SIDs in the delayed deletion state from an instance, run the **reset bgp slow-delete end-dt46 sid** command in the user view.

----End

Configuring BGP to Record Peer Status Changes and Event Information

After BGP is configured to record peer status changes, BGP records logs when the BGP peer status changes.

Context

System log files serve as an important reference for locating network connectivity and stability problems. If an error occurs on a connection between BGP peers, a corresponding error code and subcode are generated. If the local device terminates the connection with a peer due to a received a Notification message from the peer, the local device records the error code carried in the received message, and the state machine on the local device changes. When the connection is interrupted due to an error on the local end, the local end changes its state machine and sends a Notification message to the peer end.

By default, BGP records peer status changes and event information in the system log files. The record includes BGP error codes and subcodes, BGP state machine changes, and whether BGP Notification messages are sent.

If you do not want BGP to record peer status changes or event information, run the **undo peer log-change** command. After you run the **undo peer log-change** command, BGP records only the last peer status change in the log file. To check this log, run the **display bgp peer loginfo** command.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **peer { ipv4-address | group-name } log-change**

BGP is configured to record peer status changes and event information.

Step 4 Run **commit**

The configuration is committed.

----End

Disabling the PAF Restriction for the Feature Indicating Whether the Number of Received BGP Routes Exceeds the Upper Limit

You can disable the PAF restriction for the feature indicating whether the number of routes received from all peers in a BGP address family exceeds the upper limit. This configuration allows a device to continue to receive routes even after the number exceeds the upper limit.

Usage Scenario

By default, the PAF restriction is enabled for the feature indicating whether the number of routes received from all peers in a BGP address family exceeds the upper limit. With the PAF restriction, if the number of received routes exceeds 80% of the upper limit, a threshold alarm is generated. If the number exceeds the upper limit, a threshold-crossing alarm is generated, and the excess routes are discarded. To enable the local device to continue to receive routes even after the number exceeds the upper limit, run the **bgp paf feature off** command to disable the PAF restriction for this feature.

 CAUTION

Currently, disabling the PAF restriction is supported only for the feature indicating whether the number of routes received from all peers in a BGP address family exceeds the upper limit. If the PAF restriction is disabled on a BGP device, the device can still learn routes when the upper limit is exceeded, but this may consume excessive memory and affect other services. In addition, the device may be forced to restart if the memory is depleted. Therefore, disabling the PAF restriction is not recommended.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp paf feature *featureName* off**

The PAF restriction is disabled for the feature indicating whether the number of routes received from all peers in a BGP address family exceeds the upper limit.

Step 3 Run **commit**

The configuration is committed.

----End

Configuring a Mode in Which BGP Path Attributes Are Processed

To enhance reliability, you can configure a special mode in which BGP path attributes are processed.

Usage Scenario

A BGP Update message contains various path attributes. If a local device receives Update messages containing malformed path attributes, the involved BGP sessions may flap. To enhance reliability, you can perform this task to configure a mode for the device to process specified BGP path attributes.

 NOTE

This function takes effect immediately for the routes received after the **bgp path-attribute** or **peer path-attribute-treat** command is run. However, this function does not take effect immediately for the routes received before the **bgp path-attribute** or **peer path-attribute-treat** command is run. To allow this function to take effect in this case, you need to run the **refresh bgp** command.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Configure a way to handle incorrect path attributes as required.

- To allow the device to accept the path attributes with the value of 0, run the **bgp path-attribute { originator-id | attr-set } accept-zero-value** command.

- To allow the device to accept the path attributes with the length of 0, run the **bgp path-attribute { community | ext-community | ipv6-ext-community | large-community | attr-set | wide-community | clust-list } accept-zero-length** command.

The **bgp path-attribute accept-zero-value** command allows the path attributes with the value of 0 to be accepted. The **bgp path-attribute accept-zero-length** command allows the path attributes with the length of 0 to be accepted.

After the **bgp path-attribute attr-set accept-zero-value** command is run, if the Originator_ID in the Attr_Set attribute is 0, the corresponding route is accepted.

After the **bgp path-attribute attr-set accept-zero-length** command is run, if the length of the Community, Ext-community, IPv6 ext-community, Large-community, Wide-community, or Cluster_List attribute in the Attr_Set attribute is 0, the routes corresponding to the Attr_Set attribute are accepted.

NOTE

The **bgp path-attribute** command takes effect for all address families. To configure the device to perform special processing on path attributes in a specific address family, perform the following steps.

The **bgp path-attribute accept-zero-value** command takes effect for all address families. To configure the device to perform special processing on path attributes in a specific address family, run the **peer peerIpv4Addr treat-with-error attribute-id id accept-zero-value** command. Currently, the **attribute-id id** supports only the Originator_ID attribute.

The **bgp path-attribute wide-community accept-zero-length** command does not take effect in the BGP RPD address family. After a device receives an RPD route with the Wide-Community attribute whose length is 0, the device still withdraws the route even if this command is run.

Step 3 Run **bgp as-number**

BGP is started (with the local AS number specified), and the BGP view is displayed.

Step 4 Run **peer ipv4-address as-number as-number**

The IP address of a peer and the number of the AS where the peer resides are specified.

Step 5 Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

Step 6 Run **peer peerIpv4Addr path-attribute-treat attribute-id { id [to id2] } &<1-255> { discard | withdraw | treat-as-unknown }**

A special mode in which the device processes specified path attributes is configured.

NOTE

Running this command may cause path attribute discarding and route withdrawal. Therefore, exercise caution when running this command.

If both the **bgp path-attribute** and **peer path-attribute-treat attribute-id** commands are configured, the device follows the **peer path-attribute-treat attribute-id** command.

The **path-attribute-treat** parameter specifies a path attribute processing mode, which can be any of the following ones:

- Discarding specified attributes
- Withdrawing the routes with specified attributes
- Processing specified attributes as unknown attributes

Step 7 Run **commit**

The configuration is committed.

----End

Configuring Memory Overload Control

BGP can be enabled to discard newly received routes when the device memory is in the danger state and reset the corresponding process if the device memory remains in the danger state for a long time.

Usage Scenario

If BGP keeps receiving new routes when the device memory is in the danger state, the device memory may be exhausted. As a result, the process or board is reset, or even the device is restarted. To solve this problem, you can configure the device to discard the routes newly received from IPv4, IPv6, VPNv4, VPNv6, and EVPN peers.

If BGP is enabled to discard newly received routes when the device memory is in the danger state and the device memory remains in the danger state for a long time, you can enable BGP to reset the corresponding process to try to escape from the danger state.

To prevent key services from being affected, you can configure the device not to tear down the peer relationships related to the services when the device memory is in the danger state.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp memory-usage danger-threshold { discard | auto-process | accept } new-route**

BGP is enabled to process newly received routes even when the device memory is in the danger state. The parameters are described as follows:

- **discard**: When the device memory is in the warning state, the device slows down route receiving. When the device memory in the danger state, the device discards received routes.
- **auto-process**: BGP determines whether to discard received routes based on the device's memory size and CPU type. If the memory size is greater than or equal to 8 GB and the CPU type is 64-bit, BGP discards received routes by default when the device memory in the danger state. In other cases, BGP does not discard received routes by default.
- **accept**: Received routes are not discarded.

Step 3 Run **bgp memory-usage danger-threshold escape timeout time [minute]**

BGP is configured to disconnect from its peers when the device memory is in the danger state for a long time. If the device still cannot escape from the danger state, the BGP process is reset.

 NOTE

Configuring the **bgp memory-usage danger-threshold discard new-route** command may affect route learning and compromise services. Therefore, exercise caution when running this command.

Configuring the **bgp memory-usage danger-threshold accept new-route** command may cause high memory usage. Therefore, exercise caution when running this command.

Configuring the **bgp memory-usage danger-threshold escape timeout time [minute]** command may interrupt services such as IGP and route management services. Therefore, exercise caution when running this command. You can run the **bgp memory-usage danger-threshold escape disable** command to disable BGP from attempting to escape from the danger state when the device memory is in the danger state.

Step 4 (Optional) To configure the device not to disconnect from its peers even when the device memory is in the danger state, perform the following operations:

1. Run the **bgp as-number** command to enter the BGP view.
2. Run the **peer { peerIpv4Addr | groupName } low-memory exempt** command to configure the device not to disconnect from its peers when the device memory is in danger state, preventing key services from being affected.
3. Run the **quit** command to enter the system view.

Step 5 Run **commit**

The configuration is committed.

----End

Configuring Whitelist Session-CAR for BGP

You can configure whitelist session-CAR for BGP to isolate bandwidth resources by session for BGP messages. This configuration prevents bandwidth preemption among BGP sessions in the case of a traffic burst.

Context

The function of whitelist session-CAR for BGP sets an independent CAR channel for each BGP session to ensure that the bandwidth of each BGP session is not preempted by other traffic (including traffic from other sessions of the same protocol and traffic from other protocols). When BGP messages suffer a traffic burst, you can adjust the default parameters of whitelist session-CAR for BGP if they do not meet service requirements. This ensures that BGP messages can be sent properly.

 NOTE

In normal cases, you are advised to enable whitelist session-CAR for BGP. If this function becomes abnormal or adversely affects other services, you can run the **whitelist session-car bgp disable** command to disable this function.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **whitelist session-car bgp { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value }***

Parameters of whitelist session-CAR for BGP are configured.

In normal cases, you are advised to use the default values of these parameters.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

After configuring whitelist session-CAR for BGP, verify the configuration.

- Run the **display cpu-defend whitelist session-car bgp statistics slot slot-id** command to check statistics about whitelist session-CAR for BGP on a specified interface board.

To check the statistics within a specific period of time, run the **reset cpu-defend whitelist session-car bgp statistics slot slot-id** command to clear the existing statistics about whitelist session-CAR for BGP on the specified interface board; after a certain period of time, run the **display cpu-defend whitelist session-car bgp statistics slot slot-id** command.

NOTE

The statistics about whitelist session-CAR for BGP on a specified interface board cannot be restored after being cleared. Therefore, exercise caution when you run the **reset cpu-defend whitelist session-car bgp statistics slot slot-id** command.

Configuring Whitelist Session-CAR for BMP

You can configure whitelist session-CAR for BMP to isolate bandwidth resources by session for BMP messages.

Context

The function of whitelist session-CAR for BMP sets an independent CAR channel for each BMP session to ensure that the bandwidth of each BMP session is not preempted by other traffic (including traffic of other sessions of the same protocol and traffic of other protocols). When BMP messages form a traffic burst, you can adjust the bandwidth for each BMP session in whitelist session-CAR for BMP to ensure that BMP messages can be sent to the CPU properly.

NOTE

If the function becomes abnormal or affects other services, you can run the **whitelist session-car bmp disable** command to disable whitelist session-CAR for BMP. In normal cases, you are advised to keep whitelist session-CAR for BMP enabled.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **whitelist session-car bmp { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value }** *

Parameters of whitelist session-CAR for BMP are configured.

In normal cases, you are advised to use the default values.

Step 3 Run commit

The configuration is committed.

----End

Verifying the Configuration

Run the **display cpu-defend whitelist session-car bmp statistics slot slot-id** command to check statistics about IPv4 whitelist session-CAR for BMP on a specified interface board. To check the statistics within a specific period of time, run the **reset cpu-defend whitelist session-car bmp statistics slot slot-id** command to clear the existing statistics about IPv4 whitelist session-CAR for BMP on the specified interface board; after a certain period of time, run the **display cpu-defend whitelist session-car bmp statistics slot slot-id** command.



The statistics about whitelist session-CAR for BMP on a specified interface board cannot be restored after being cleared. Therefore, exercise caution before clearing them.

Configuring Whitelist Session-CAR for RPKI

You can configure whitelist session-CAR for RPKI to isolate bandwidth resources by session for RPKI messages.

Context

The function of whitelist session-CAR for RPKI sets an independent CAR channel for each RPKI session to ensure that the bandwidth of each RPKI session is not preempted by other traffic (including traffic of other sessions of the same protocol and traffic of other protocols). When RPKI messages form a traffic burst, you can adjust the bandwidth for each RPKI session in whitelist session-CAR for RPKI to ensure that RPKI messages can be sent to the CPU properly.



If the function becomes abnormal or affects other services, you can run the **whitelist session-car rpk disable** command to disable whitelist session-CAR for RPKI. In normal cases, you are advised to keep whitelist session-CAR for RPKI enabled.

Procedure

Step 1 Run system-view

The system view is displayed.

Step 2 Run **whitelist session-car rPKI { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } ***

Parameters of whitelist session-CAR for RPKI are configured.

In normal cases, you are advised to use the default values.

Step 3 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the **display cpu-defend whitelist session-car rPKI statistics slot slot-id** command to check statistics about IPv4 whitelist session-CAR for RPKI on a specified interface board. To check the statistics within a specific period of time, run the **reset cpu-defend whitelist session-car rPKI statistics slot slot-id** command to clear the existing statistics about IPv4 whitelist session-CAR for RPKI on the specified interface board; after a certain period of time, run the **display cpu-defend whitelist session-car rPKI statistics slot slot-id** command.



The statistics about whitelist session-CAR for RPKI on a specified interface board cannot be restored after being cleared. Therefore, exercise caution before clearing them.

Configuring Micro-Isolation CAR for BGP

You can configure micro-isolation CAR for BGP to isolate bandwidth resources by interface or sub-interface for BGP messages used to establish peer relationships.

Context

When BGP messages suffer a traffic burst, bandwidth may be preempted among interfaces and sub-interfaces. As a result, the BGP messages may fail to be sent properly. To resolve this problem, you can configure micro-isolation CAR for BGP to isolate bandwidth resources by interface or sub-interface for the BGP messages. If the default parameters of micro-isolation CAR for BGP do not meet service requirements, you can adjust them as required to ensure that the BGP messages can be sent properly.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **micro-isolation protocol-car bgp { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } ***

Parameters of micro-isolation CAR for BGP are configured.

In normal cases, you are advised to use the default values of these parameters.

Step 3 (Optional) Run **micro-isolation protocol-car bgp disable**

Micro-isolation CAR for BGP is disabled.

In normal cases, you are advised to keep this function enabled. Disable it if it becomes abnormal or adversely affects other services.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.9.2.57 BGP Route Selection Rules

Route Processing on the BGP router

Understanding how the BGP router processes routes helps you better control the BGP routing table.

Figure 1-404 shows the route processing on the BGP router. BGP routes can be imported from other protocols or learned from BGP peers. Route summarization can be configured to reduce the routing table size. Then the router performs operations such as route selection, advertisement, and delivery to the IP routing table.

Figure 1-404 Route processing on the BGP router

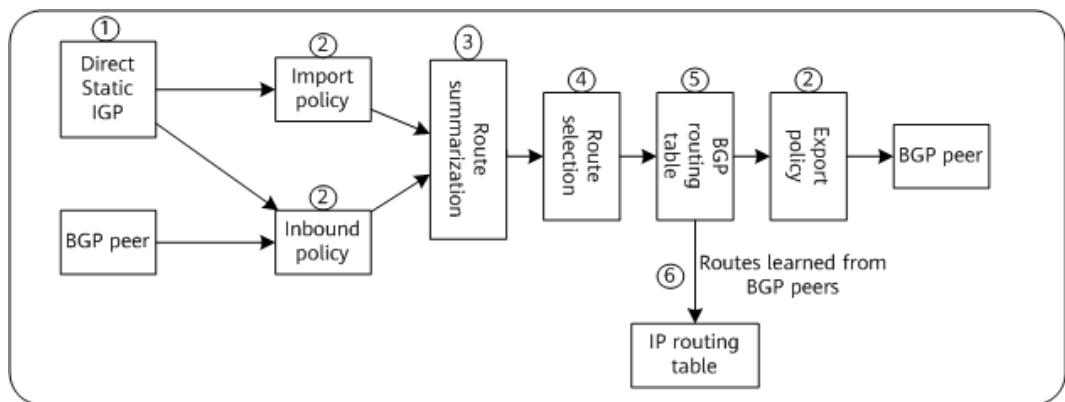


Table 1-129 lists some key points in **Figure 1-404**.

Table 1-129 Key points for route processing

No.	Remarks
1	BGP can import direct routes, static routes, user network routes, and IGP routes based on the import-route (BGP) or network command configuration.
2	BGP can use routing policies when importing routes from other protocols, receiving routes from BGP peers, or advertising routes to BGP peers. Routing policies can be used to filter routes or modify route attributes.

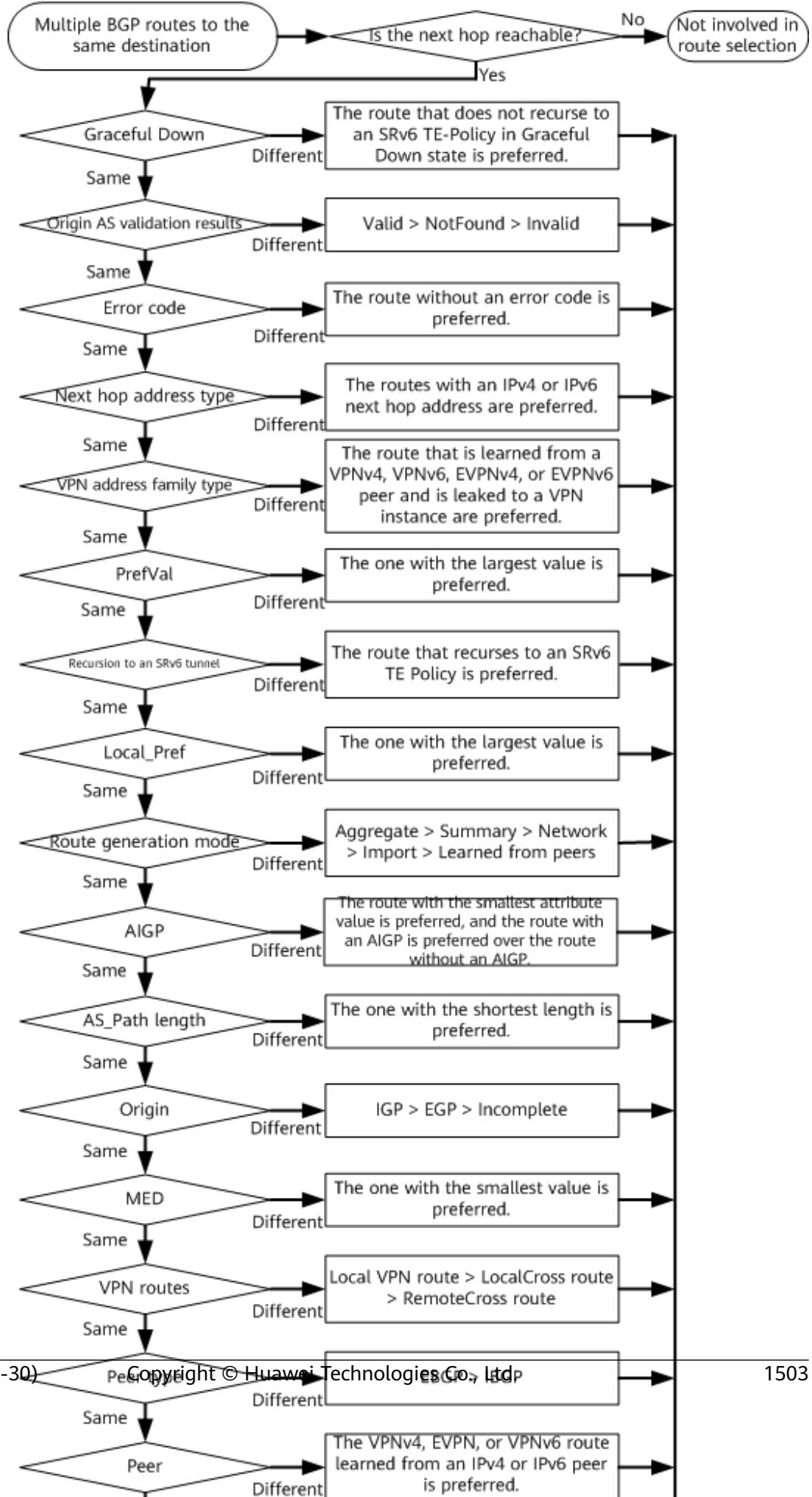
No.	Remarks
3	BGP supports automatic and manual summarization. Multiple routing policies can be used during manual summarization.
4	BGP selects routes based on strict route selection rules, which is the key point to be discussed in the following part.
5	BGP adds the optimal route to the BGP routing table and advertises it to BGP peers.
6	BGP adds the routes learned from peers and the optimal route in the BGP routing table to the IP routing table for traffic forwarding.

Route Selection Rules

When multiple received routes are available to the same destination, BGP selects one optimal route based on BGP route selection rules and adds it to the IP routing table for traffic forwarding.

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, when multiple routes to the same destination are available, BGP selects according to [Figure 1-405](#). For details about route selection rules, see [BGP Route Processing](#).

Figure 1-405 BGP route selection process



BGP selects routes by comparing route attributes in a fixed order. When a route attribute is a sufficient condition for determining the optimal route, BGP does not compare the other attributes. If BGP fails to select the optimal route after comparing all route attributes, the route that was first received is selected as the optimal route. **Table 1-130** lists the abbreviated alias, route selection rules, and remarks of each matching item. **Table 1-130** shows that the route priority is directly proportional to the PrefVal or Local_Pref value and inversely proportional to the rest of the attribute values or lengths. In addition, the first column can be summarized as a character string (OPPAAA OMTCC RA), which helps memorize the matching sequence.

Table 1-130 BGP route selection process

Abbr eviated Alias	Matching Item	Route Selection Rules	Remarks
O	Origin AS	Valid > NotFound > Invalid	BGP origin AS validation states are applied to route selection in a scenario where the device is connected to an RPKI server.
P	PrefVal	The route with the largest PrefVal value is preferred. The default value is 0.	It is valid only locally.
P	Local_Pref	The route with the largest Local_Pref value is preferred. The default value is 100.	To modify the default Local_Pref value of BGP routes, run the default local-preference command.

Abbr eviated Alias	Matching Item	Route Selection Rules	Remarks
A NOTE A is the initial of the character string (AS NIL).	Route generation mode	<p>A > S > N > I > L, where:</p> <ul style="list-style-type: none"> • A: indicates that routes are summarized using the aggregate command. • S: indicates that routes are summarized using the summary automatic command. • N: indicates that routes are imported using the network command. • I: indicates that routes are imported using the import-route command. • L: indicates that routes are learned from BGP peers. 	-
A	Accumulated Interior Gateway Protocol (AIGP)	<p>The route with the smallest AIGP value is preferred.</p> <p>The route with AIGP to the route without AIGP is preferred.</p>	-
A	AS_Path	The route with the shortest AS_Path length is preferred.	If the bestroute as-path-ignore command is run, BGP does not compare the AS_Path lengths in different routes during route selection.
O	Origin	IGP > EGP > Incomplete	-

Abbr eviated Alias	Matching Item	Route Selection Rules	Remarks
M	Multi Exit Discriminator (MED)	The route with the smallest MED value is preferred. The default value is 0.	If the bestroute med-none-as-maximum command is configured, BGP considers the largest MED value (4294967295) as the MED of the route that does not carry an MED. For details about MED usage, see MED .
T	Peer type	EBGP > IBGP	-
C	IGP metric	The route with the smallest IGP cost is preferred.	If the bestroute igr-metric-ignore command is configured, BGP does not compare the IGP cost.
C	Cluster_List	The route with the shortest Cluster_List length is preferred.	By default, BGP compares Cluster_Lists prior to Originator_IDs during route selection. To enable BGP to compare Originator_IDs prior to Cluster_Lists during route selection, run the bestroute routerid-prior-clusterlist command.
R	Router ID	The route with the smallest router ID is preferred.	If routes carry the Originator_ID, the originator ID is substituted for the router ID during route selection. The route with the smallest Originator_ID is preferred.
A	Peer IP address	The route learned from the peer with the smallest IP address is preferred.	-

Selection of the Routes for Load Balancing

After BGP load balancing is configured, the BGP routes that meet the following conditions are used as equal-cost routes for load balancing:

- Original next-hop addresses are different.
- The routes have the same PrefVal value.
- The routes have the same Local_Pref value.

- All the routes are summarized or non-summarized routes.
- The routes have the same AIGP value.
- The routes have the same AS_Path length.
- The routes have the same origin type (IGP, EGP, or incomplete).
- The routes have the same MED value.
- All the routes are EBGP or IBGP routes. If the **maximum load-balancing eibgp** command is run, load balancing can be implemented among EBGP and IBGP routes.
- The metric values of the IGP routes to which BGP routes within an AS recurse are the same. After the **load-balancing igrp-metric-ignore** command is run, the device does not compare IGP metric values when selecting routes for load balancing.
- All routes are blackhole routes, or none of them are blackhole routes.

Load balancing can be implemented between non-leaked routes or between leaked routes, but not between labeled BGP routes and non-labeled BGP routes even if the preceding conditions are met. If the **load-balancing local-learning cross** command is run in a VPN, load balancing can be implemented between unlabeled unicast VPN routes and labeled routes. Load balancing cannot be implemented between blackhole routes and non-blackhole routes.

VPN Route Selection Rules

On the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X, the rules for selecting VPN BGP routes are the same as those for selecting public network BGP routes. The only difference is that VPN BGP routes need to be leaked based on VPN targets. For details about route leaking, see "BGP VPN Route Leaking" in NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X *Feature Description > IP Routing > BGP*.

BGP Routing Table

This section describes how to check route attributes.

Table 1-131 lists all the common route attributes that affect route selection and the commands that are used to check them.

Table 1-131 Commands used to check route attributes

Route Attribute	Command Used to Check the Route Attribute
Origin AS	display bgp routing-table [network]
PrefVal	display bgp routing-table [network]
Local_Pref	display bgp routing-table [network]
Route type	display bgp routing-table network
AIGP	display bgp routing-table network
AS_Path	display bgp routing-table [network]

Route Attribute	Command Used to Check the Route Attribute
Origin	display bgp routing-table [network]
MED	display bgp routing-table [network]
Peer type	display bgp routing-table network
IGP Metric	<ul style="list-style-type: none"> • display bgp routing-table network • display ip routing-table ip-address [mask mask-length] [verbose], in which <i>ip-address</i> is the next hop IP address of a BGP route
Cluster_List	display bgp routing-table network
Originator_ID	display bgp routing-table network
Router ID	display bgp routing-table network
Peer IP address	display bgp routing-table network

The following example describes how to check BGP route attributes in the **display bgp routing-table** command output.

```
<HUAWEI> display bgp routing-table
BGP Local router ID is 1.1.1.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete  RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 4
      Network     NextHop     MED     LocPrf   PrefVal Path/Ogn
* 10.1.1.0/24  10.1.1.1    0        0        100?
* 10.1.1.2/32  10.1.1.1    0        0        100?
*> 10.1.1.0/24  10.1.1.1    0        0        100?
*> 10.10.1.0/24 10.1.1.1    0        0        100?
```

Table 1-132 Description of the **display bgp routing-table** command output

Item	Description
BGP Local router ID is 1.1.1.2	Router ID: 1.1.1.2, in the same format as an IPv4 address

Item	Description
Status codes	<p>Route status code, displayed in front of each route entry:</p> <ul style="list-style-type: none"> ● *: indicates a valid route with a reachable next hop address. ● >: indicates an optimal route selected by BGP. ● d: indicates a dampened route. ● x: indicates a Best-external route. ● a: indicates an ADD-PATH route. ● h: indicates a History route. ● i: indicates a route learned from an IBGP peer. ● s: indicates a suppressed route. If specific routes for route summarization are suppressed, s is displayed in front of each specific route. ● S: indicates a route in Stale status, and the route is being deleted. Such routes may occur during a BGP GR process. <p>BGP dampening measures route stability using a penalty value. The greater the penalty value, the less stable a route. Each time route flapping occurs (a device receives a Withdraw or an Update message), BGP adds a penalty value to the route carried in the message.</p> <p>When the penalty value of a route exceeds the Suppress value, BGP removes the > flag from the route. Then, the route is suppressed and does not participate in BGP route selection, and the router does not advertise Update messages of this route to other BGP peers.</p> <ul style="list-style-type: none"> ● If d is displayed in front of a route, the route is carried in an Update packet. ● If h is displayed in front of a route, the route is carried in a Withdraw packet. <p>The penalty value is not increased after it reaches the suppression threshold. The penalty value of a suppressed route reduces by half after a half-life period.</p> <ul style="list-style-type: none"> ● When the penalty value of a route with the d sign decreases to the Reuse value, the route becomes reusable, and BGP removes the d sign, adds the route to the IP routing table, and advertises an Update packet carrying the route to BGP peers. ● When the penalty value of a route with the h sign decreases to 0, BGP deletes this route from the BGP routing table.
Origin	<p>Route Origin:</p> <ul style="list-style-type: none"> ● IGP: indicates that routes are added to the BGP routing table using the network (BGP) command. ● EGP: indicates that routes are learned through the EGP protocol. ● Incomplete: indicates that routes are added to the BGP routing table using the import-route (BGP) command.

Item	Description
RPKI validation codes	Route origin AS validation code. <ul style="list-style-type: none"> • V: indicates a valid route. • I: indicates an invalid route. • N: indicates a not-found route.
Network	Network address in the BGP routing table
NextHop	Next hop address
MED	MED value of a BGP route, similar to the cost of IGP routes
LocPrf	Local_Pref
PrefVal	PrefVal
Path/Ogn	AS_Path and Origin attributes

Information about Next_Hop, MED, Local_Pref, PrefVal, AS_Path, and Origin can be displayed using the **display bgp routing-table** command. To check information about the route type, AIGP, peer type, IGP cost, Cluster_List, router ID, and peer IP address, run the **display bgp routing-table network** command.

```
<HUAWEI> display bgp routing-table 10.1.1.1
BGP local router ID : 192.168.2.2
Local AS number : 100
Paths: 1 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 10.1.1.1/32:
From: 10.1.3.1 (192.168.2.3)
Route Duration: 0d00h01m33s
Direct Out-interface: GigabitEthernet0/1/0
Relay is delayed as nexthop flapped frequently Original nexthop: 10.1.3.1
Qos information : 0x0
Primary Routing Table: vrf1 AS-path 200, origin incomplete, MED 0, pref-val 0, valid, external, best, select,
active, pre 255, IGP cost 20
Advertised to such 1 peers:
    10.1.3.1
```

Table 1-133 Description of the **display bgp routing-table** command output

Item	Description
BGP local router ID	Router ID of the local device, in the same format as an IPv4 address.
Local AS number	Local AS number.
Paths	BGP route information.
BGP routing table entry information of 10.1.1.1/32	Information about the BGP route 10.1.1.1/32:

Item	Description
From	IP address of the device that sends the route. 10.1.3.1 is the IP address (Peer IP Address) of the interface used by the peer to set up a BGP connection, and 192.168.2.3 is the router ID of the peer.
Route Duration	Duration of a route.
Direct Out-interface	Directly connected interface.
Relay is delayed as nexthop flapped frequently	Route recursion to a specified next hop is suppressed because the next hop flaps. If only a small number of routes recurse to the next hop, the suppression is very short; therefore, this field may not be displayed in this case.
Original nexthop	Original next hop IP address.
Qos information	QoS information.
Primary Routing Table	The source routing table
AS-path	AS_Path attribute. If Nil is displayed, the AS_Path attribute is null.
origin incomplete	Origin attribute of the route, which can be any of the following three values: <ul style="list-style-type: none"> • IGP: indicates that routes are added to the BGP routing table using the network (BGP) command. • EGP: indicates that routes are learned through the EGP protocol. • Incomplete: indicates that routes are imported using the import-route (BGP) command.
MED	MED value of a BGP route, similar to the cost of IGP routes.
pref-val	PrefVal
valid	Valid route with a reachable next hop address.
external	Type of the peer from which the route is learned. <ul style="list-style-type: none"> • external: indicates that the route is learned from an EBGP peer. • internal: indicates that the route is learned from an IBGP peer.
best	Optimal route.

Item	Description
select	Selected route to be delivered to the IP routing table. NOTE According to BGP selection rules, BGP selects only one optimal route, and this route is marked with best . In load balancing or FRR scenarios, more than one route needs to be added to the IP routing table, and each of the route is marked with select . Therefore, the number of the route marked with best is 1, and the number of the routes marked with select is the actual number of BGP routes added to the IP routing table.
active	Active route.
pre 255	Protocol priority of the route: 255
Advertised to such 1 peers	The BGP route has been advertised to one peer.

The **display bgp routing-table network [{ mask | mask-length } [longer-prefixes]]** command output is related to the route generation and transmission modes. Not all attributes of BGP routes are displayed. In the preceding command output, the route type of the route 10.1.1.1/32 is not displayed because it is an IBGP route. If you run the **display bgp routing-table network [{ mask | mask-length } [longer-prefixes]]** command, the route type will be displayed. For example:

```
<HUAWEI> display bgp routing-table 10.0.0.0
BGP local router ID : 192.168.2.4
Local AS number : 200
Paths: 1 available, 1 best, 1 select
BGP routing table entry information of 10.0.0.0/8:
Aggregated route.
Route Duration: 04h50m46s
Direct Out-interface: NULL0
Original nexthop: 127.0.0.1
Qos information : 0x0
AS-path {65001 10 100}, origin incomplete, pref-val 0, valid, local, best, select, active, pre 255
Aggregator: AS 200, Aggregator ID 192.168.2.4, Atomic-aggregate
Advertised to such 3 peers:
    10.1.7.2
    172.16.1.2
    192.168.1.2
```

The route 10.0.0.0/8 was manually summarized using the **aggregate** command. Therefore, **Aggregated route** is displayed in the command output. The route type varies as follows:

- If the route is automatically summarized using the **summary automatic** command, **Summary automatic route** will be displayed.
- If the route is imported using the **network** command, **Network route** will be displayed.
- If the route is imported using the **import-route** command, **Imported route** will be displayed.

In the following example, an RR and a cluster are configured. Therefore, the Cluster_List attribute is displayed in the **display bgp routing-table network [{ mask | mask-length } [longer-prefixes]]** command output. For example:

```
<HUAWEI> display bgp routing-table 10.2.1.0
BGP local router ID : 4.4.4.4
Local AS number : 65010
Paths: 1 available, 0 best, 0 select
BGP routing table entry information of 10.2.1.0/24:
From: 10.1.4.1 (2.2.2.2)
Route Duration: 00h00m14s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface:
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path Nil, origin igp, MED 0, localpref 100, pref-val 0, internal, pre 255
Originator: 1.1.1.1
Cluster list: 0.0.0.1
Not advertised to any peer yet
```

Route Attributes

?1. BGP Next_Hop

BGP ignores routes with an unreachable next hop address during BGP route selection.

Unlike the Next_Hop attribute in an IGP, the Next_Hop attribute in BGP is not necessarily the IP address of a neighboring device. In most cases, the Next_Hop attribute in BGP complies with the following rules:

- When advertising a route to an EBGP peer, the BGP speaker sets the Next_Hop of the route to the address of the local interface through which the BGP peer relationship is established.
- When advertising a locally generated route to an IBGP peer, the BGP speaker sets the Next_Hop of the route to the address of the local interface through which the BGP peer relationship is established.
- When advertising a route learned from an EBGP peer to an IBGP peer, the BGP speaker does not change the Next_Hop of the route.
- When advertising a route learned from an IBGP peer to another IBGP peer, the BGP speaker does not change the Next_Hop of the route.

Modifying the Next_Hop

In some scenarios, the Next_Hop needs to be modified. [Table 1-134](#) describes whether the Next_Hop needs to be modified in specific scenarios.

Table 1-134 Next_Hop processing

Objectives	Command	Usage Scenarios	Remarks
To enable the device to modify the Next_Hop of the routes to be advertised to an IBGP peer	peer { ipv4-address group-name } next-hop-local	By default, a device does not change the next hop address of a route learned from an EBGP peer before forwarding the route to IBGP peers. The next hop address of a route advertised by an EBGP peer to this device is the peer address of the EBGP peer. After being forwarded to IBGP peers in the local AS, this route is not active because the next hop is unreachable. To address this problem, the relevant ASBR needs to be configured to change the Next_Hop of the route learned from an EBGP peer to the ASBR's own address before the ASBR advertises the route to an IBGP peer. As an IGP runs within the AS, the next hop of the route is reachable to the IBGP peer. As such, the route received by the IBGP peer is active.	NOTE If BGP load balancing is configured using the maximum load-balancing number command, the router changes the next hop address of a route to the IP address used to establish an IBGP peer relationship when advertising the route to the IBGP peer or peer group, regardless of whether the peer next-hop-local command is run.
To configure a device to retain the original Next_Hop of imported IGP routes when the device advertises the routes to an IBGP peer	peer { ipv4-address group-name } next-hop-invariable	In an intra-AS scenario, if a device is configured to retain the original Next_Hop of imported IGP routes when advertising the routes to an IBGP peer, the peer can use the original Next_Hop for recursion, which reduces the number of hops.	-

Objectives	Command	Usage Scenarios	Remarks
To configure a BGP device to retain the original Next_Hop of imported static routes when advertising the routes to an IBGP peer	peer { ipv4-address group-name } next-hop-invariable include-static-route	In an intra-AS scenario, if a device is configured to retain the original Next_Hop of imported static routes when advertising the routes to an IBGP peer, the peer can use the original Next_Hop for recursion, which reduces the number of hops.	-
To configure a BGP device to retain the original Next_Hop when the device advertises to an EBGP peer the unicast routes learned from another peer	peer { ipv4-address group-name } next-hop-invariable include-unicast-route	In an intra-AS scenario, if a device is configured to retain the original Next_Hop of unicast routes when advertising them to a peer, the peer can directly use the original Next_Hop for recursion, which reduces the number of hops.	-

Objectives	Command	Usage Scenarios	Remarks
To prevent a device from modifying the Next_Hops of routes before advertising the routes to an EBGP peer	peer { ipv4-address group-name } next-hop-invariable	In an inter-AS VPN Option C scenario where RRs are used, the peer next-hop-invariable command needs to be run on the RRs to prevent them from modifying the Next_Hops of routes before advertising the routes to an EBGP peer. This ensures that the remote PE can implement recursion to the BGP LSP to the local PE during traffic transmission.	- By default, a BGP device changes the Next_Hop of routes to its local interface IP address before advertising the routes to EBGP peers. In addition, a BGP device does not modify the Next_Hop of unlabeled routes if the routes are learned from EBGP peers and are to be advertised to IBGP peers; however, the device changes the Next_Hop to its own interface IP address if these routes are labeled routes.
To configure route-policy-based next hop recursion	nexthop recursive-lookup route-policy route-policy-name	Route-policy-based next hop recursion helps flexibly control the recursion result based on specific conditions. If a route does not match the specified route-policy, the route fails to recurse.	-

Objectives	Command	Usage Scenarios	Remarks
To enable a device to modify the Next_Hops of BGP routes using a route-policy	apply ip-address next-hop { ipv4-address peer-address }	<p>The Next_Hops of BGP routes can be modified using a route-policy in the following situations:</p> <ul style="list-style-type: none"> For an IBGP peer, the route-policy can be an import or export policy. Even if the next hop address configured in the route-policy is unreachable, the IBGP peer still adds the routes whose next hop addresses have been changed to the address configured in the route-policy to the BGP routing table. However, the routes are invalid. For an EBGP peer, to modify the next hop address of routes, an import policy is configured in most cases. If the route-policy is configured as an export policy, the routes whose next hop addresses have been changed to the address configured in the route-policy are discarded by the EBGP peer because the next hop address is unreachable. 	If a route-policy has been specified in the import-route or network command, the apply clause configured for the route-policy using the apply ip-address next-hop command does not take effect.

Obtaining a Reachable BGP Next Hop

During route selection, BGP first checks whether the next hop addresses of routes are reachable. Routes carrying unreachable next hop addresses are inactive and

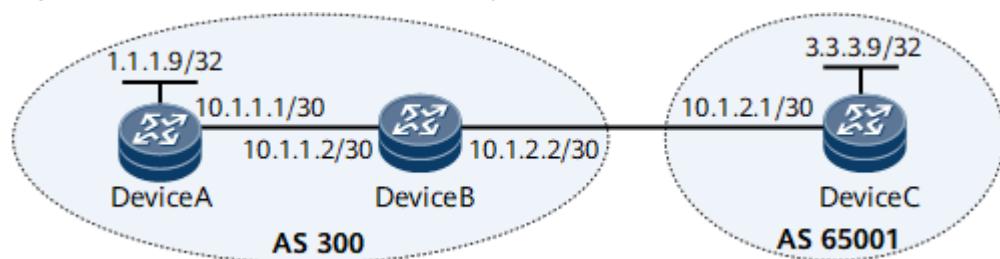
are not selected. As described in [Table 1-135](#), the next hop is unreachable in the following situations:

Table 1-135 Unreachable next hop

Item	Description	Solutions
Unreachable next hop IP address	A next hop IP address is obtained through route recursion, but no active routes to the IP address are available in the IP routing table.	Common solutions are as follows: <ul style="list-style-type: none">• Configure static routes or an IGP.• Run the import-route command.• Run the network command. Alternatively, you can run the peer next-hop-local command to change the Next_Hop to the local IP address.
Unreachable next hop tunnel	Routes fail to recurse to tunnels.	Configure a tunnel policy or a tunnel selector to ensure that the routes can recurse to tunnels.
	A next hop tunnel is obtained through route recursion, but the tunnel is unavailable.	Ensure that the tunnel is correctly configured and is Up.

[Figure 1-406](#) is used to show how to obtain a reachable next hop IP address. In [Figure 1-406](#), an IBGP peer relationship is established between DeviceA and DeviceB, and an EBGP peer relationship is established between DeviceB and DeviceC. DeviceA imports the route 1.1.1.9/32, and DeviceC imports the route 3.3.3.9/32.

Figure 1-406 BGP route unreachability



Display the BGP routing table of DeviceA.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
```

h - history, i - internal, s - suppressed, S - Stale Origin : i - IGP, e - EGP, ? - incomplete RPKI validation codes: V - valid, I - invalid, N - not-found						
Total Number of Routes: 2						
Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn	
*> 1.1.1.9/32	0.0.0.0	0	0	i		
i 3.3.3.9/32	10.1.2.1	0	100	0	65001i	

The preceding command output shows that no asterisk (*) is in front of the route 3.3.3.9/32, which indicates that the route is invalid.

Display the IP routing table of DeviceA.

[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Destinations : 5 Routes : 5
Destination/Mask Proto Pre Cost Flags NextHop Interface
1.1.1.9/32 Direct 0 0 D 127.0.0.1 LoopBack1
10.1.1.0/30 Direct 0 0 D 10.1.1.1 GigabitEthernet1/0/0
10.1.1.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0

The preceding command output shows that the next hop 10.1.2.1 of the route 3.3.3.9/32 is not in the IP routing table. This indicates that the route 3.3.3.9/32 becomes invalid because its next hop is unreachable. The following methods can be used for the route 3.3.3.9/32 to become valid:

- Configure a static route destined for 10.1.2.1/30 on DeviceA.
- Configure an IGP on DeviceB and DeviceC and configure BGP on DeviceB to import the route 10.1.2.1. However, this method is not applicable to this scenario because DeviceB and DeviceC are located in different ASs.
- Run the **import-route direct** command on DeviceB. This solution is not optimal because unnecessary routes may be imported.
- Run the **network 10.1.2.0 30** command on DeviceB to configure BGP to advertise the route 10.1.2.0/30 to DeviceA.
- Run the **peer 10.1.1.1 next-hop-local** command on DeviceB to configure DeviceB to modify the Next_Hop of the route 3.3.3.9/32 before advertising the route to DeviceA.

In this example, the **network 10.1.2.0 30** command is run on DeviceB. After the command is run, check the BGP routing table of DeviceA.

[~DeviceA] display bgp routing-table					
BGP Local router ID is 10.1.1.1					
Status codes: * - valid, > - best, d - damped, x - best external, a - add path, h - history, i - internal, s - suppressed, S - Stale					
Origin : i - IGP, e - EGP, ? - incomplete					
RPKI validation codes: V - valid, I - invalid, N - not-found					
Total Number of Routes: 3					
Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
*> 1.1.1.9/32	0.0.0.0	0	0	i	
*>i 3.3.3.9/32	10.1.2.1	0	100	0	65001i
*>i 10.1.2.0/30	10.1.1.2	0	100	0	i

The preceding command output shows that both * and > are in front of the route 3.3.3.9/32, which indicates that the route is valid and optimal.

?.2. PrefVal

BGP prefers the route with the largest PreVal value during BGP route selection.

PrefVal is valid only on the local device and is not transmitted to BGP peers. PrefVal values are manually set and represent the intention of the local user. Therefore, BGP preferentially compares PrefVal values during route selection.

To configure a PreVal value for the routes learned from a peer or peer group, run the **peer { group-name | ipv4-address } preferred-value value** command.

If multiple routes are available to the same destination, the route with the largest PreVal value is selected as the optimal route.

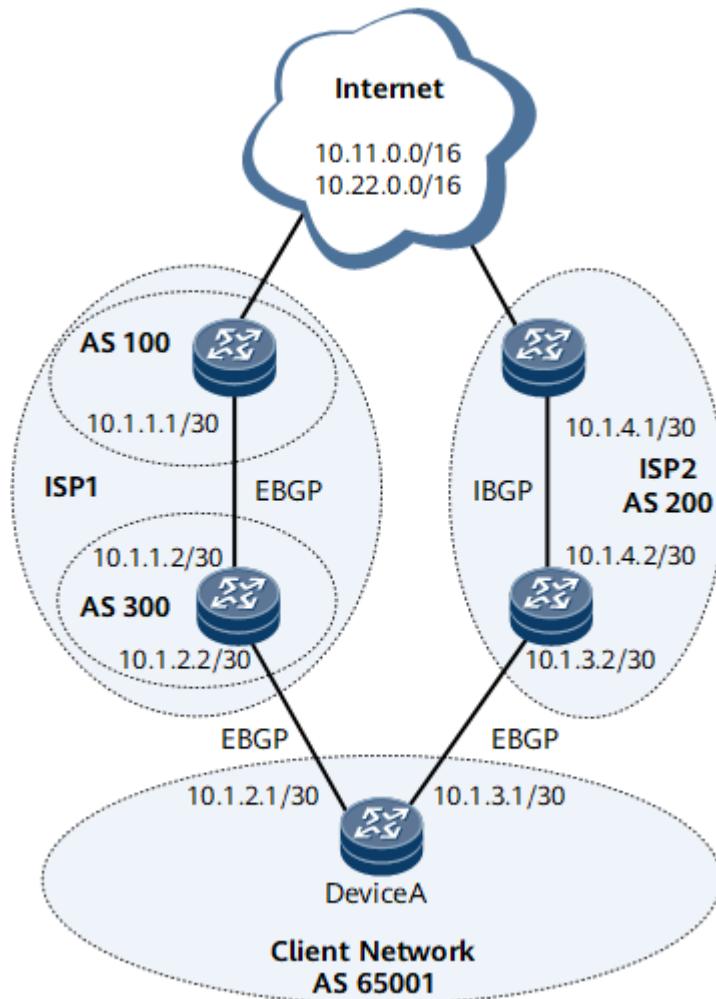
Table 1-136 lists two methods to modify the PreVal value.

Table 1-136 Methods to modify the PreVal value

Method	Usage Scenario
Run the peer { group-name ipv4-address } preferred-value value command.	This method sets a PreVal value for the routes learned from a peer or peer group.
Configure an import policy and run the apply preferred-value preferred-value command to configure an apply clause for the policy.	This method sets different PreVal values for different routes learned from a peer or peer group. NOTE If a route matches both the peer preferred-value and apply preferred-value commands, the apply preferred-value command takes effect.

The following example shows how the PreVal value is used during route selection. In [Figure 1-407](#), both ISP1 and ISP2 advertise the routes 10.11.0.0/16 and 10.22.0.0/16 to AS 65001.

Figure 1-407 PreVal application networking



Scenario 1: When no PreVal value is configured on Device A, check the BGP routing table of Device A.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 4
      Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
*>  10.11.0.0/16    10.1.3.2          0      200??
*           10.1.2.2          0      300 100??
*>  10.22.0.0/16    10.1.3.2          0      200??
*           10.1.2.2          0      300 100??
```

The BGP routing table of Device A shows that Device A receives the routes 10.11.0.0/16 and 10.22.0.0/16 from ISP1 and ISP2. Check the information about the route 10.11.0.0/16 on Device A.

```
[~DeviceA] display bgp routing-table 10.11.0.0
BGP local router ID : 10.1.2.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
```

```
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.3.2 (10.1.3.2)
Route Duration: 00h08m35s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 200, origin incomplete, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.3.2
    10.1.2.2
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 00h04m38s
Direct Out-interface: 1/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 300 100, origin incomplete, pref-val 0, valid, external, pre 255, not preferred for AS-Path
Not advertised to any peer yet
```

The preceding command output shows that the AS_Path of the route learned from ISP2 is shorter than that of the route learned from ISP1. Therefore, the route learned from ISP2 is selected as the optimal route. **Table 1-137** shows the route attribute comparison of the routes learned from ISP1 and ISP2.

Table 1-137 Route attribute comparison of the route learned from ISP1 and that learned from ISP2

Route Attribute	Route Learned from ISP1	Route Learned from ISP2	Comparison
PrefVal	0	0	The same.
Local_Pref	-	-	The same. NOTE If a route does not carry Local_Pref, the default value 100 takes effect.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	300 100	200	The route learned from ISP2 is selected as the optimal route because its AS_Path is shorter than that of the route learned from ISP1.

Scenario 2: The administrator of AS 65001 requires that ISP1 be active and ISP2 be backup for the traffic to 10.11.0.0/16 and 10.22.0.0/16.

To meet the preceding requirements, run the **peer { group-name | ipv4-address } preferred-value value** command on DeviceA to increase the PrefVal values of the

routes learned from AS 300. This configuration ensures that the routes learned from ISP1 are selected by DeviceA as the optimal routes to 10.11.0.0/16 and 10.22.0.0/16. Detailed configurations are as follows:

```
bgp 65001
#
ipv4-family unicast
peer 10.1.2.2 preferred-value 120 //Set the PrefVal of the routes learned from AS 300 to 120.
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device A.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
               h - history, i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 4
      Network          NextHop        MED     LocPrf  PrefVal Path/Ogn
* > 10.11.0.0/16    10.1.2.2           120    300 100?
*          10.1.3.2           0    200?
*> 10.22.0.0/16    10.1.2.2           120    300 100?
*          10.1.3.2           0    200?
```

The preceding command output shows that Device A selects the routes learned from ISP1.

Display detailed information about the route 10.11.0.0/16 or 10.22.0.0/16 on Device A. The route 10.11.0.0/16 is used as an example.

```
[~DeviceA] display bgp routing-table 10.11.0.0
BGP local router ID : 10.1.2.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 00h05m36s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 300 100, origin incomplete, pref-val 120, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
  10.1.3.2
  10.1.2.2
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.3.2 (10.1.3.2)
Route Duration: 00h23m11s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 200, origin incomplete, pref-val 0, valid, external, pre 255, not preferred for PreVal
Not advertised to any peer yet
```

The preceding command output shows that the PrefVal value of the route learned by DeviceA from ISP1 is larger and that the route learned from ISP1 is selected as the optimal route.

Scenario 3: The expected configurations of the administrator of AS 65001 are as follows:

- For the traffic destined to 10.11.0.0/16, ISP1 is active and ISP2 is backup.

- For the traffic destined to 10.22.0.0/16, ISP2 is active and ISP1 is backup.

To meet the preceding requirements, ensure that the route 10.11.0.0/16 learned from ISP1 and the route 10.22.0.0/16 learned from ISP2 are selected in AS 65001. In this situation, the **peer preferred-value** command can no longer be used because different PrefVal values are required for the routes learned from the same peer. In this case, configure import policies. Detailed configurations are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
  peer 10.1.2.2 route-policy for_isp1_in import      //Apply import policy named for_isp1_in to the routes learned from 10.1.2.2 and use for_isp1_in to modify the PrefVal value.  
  peer 10.1.3.2 route-policy for_isp2_in import      //Apply import policy named for_isp2_in to the routes learned from 10.1.3.2 and use for_isp2_in to modify the PrefVal value.  
#  
route-policy for_isp1_in permit node 10          //Define the first node of for_isp1_in and set the PrefVal value of the route 10.11.0.0/16 to 80.  
  if-match ip-prefix for_isp1  
    apply preferred-value 80  
#  
route-policy for_isp1_in permit node 20          //Define the second node of for_isp1_in and allow for_isp1_in to permit all routes.  
#  
route-policy for_isp2_in permit node 10          //Define the first node of for_isp2_in and set the PrefVal value of the route 10.22.0.0/16 to 120.  
  if-match ip-prefix for_isp2  
    apply preferred-value 120  
#  
route-policy for_isp2_in permit node 20          //Define the second node of for_isp2_in and allow for_isp2_in to permit all routes.  
#  
ip ip-prefix for_isp1 index 10 permit 10.11.0.0 16 //Configure an IP prefix list to match the route 10.11.0.0/16.  
ip ip-prefix for_isp2 index 10 permit 10.22.0.0 16 //Configure an IP prefix list to match the route 10.22.0.0/16.  
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device A

```
[~DeviceA] display bgp routing-table  
BGP Local router ID is 10.1.2.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
              h - history, i - internal, s - suppressed, S - Stale  
              Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 4  
  Network        NextHop       MED     LocPrf  PrefVal Path/Ogn  
*>  10.11.0.0/16   10.1.2.2           80     300 100?  
*    10.1.3.2          0     200?  
*>  10.22.0.0/16   10.1.3.2          120     200?  
*    10.1.2.2          0     300 100?
```

The preceding command output shows that Device A selects the route 10.11.0.0/16 learned from ISP1 and the route 10.22.0.0/16 learned from ISP2.

Display detailed information about the route 10.22.0.0/16 on Device A.

```
[~DeviceA] display bgp routing-table 10.22.0.0
```

```
BGP local router ID : 10.1.2.1  
Local AS number : 65001
```

```
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.22.0.0/16:
From: 10.1.3.2 (10.1.3.2)
Route Duration: 00h14m14s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 200, origin incomplete, pref-val 120, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.3.2
    10.1.2.2
BGP routing table entry information of 10.22.0.0/16:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 00h07m54s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 300 100, origin incomplete, pref-val 0, valid, external, pre 255, not preferred for PreVal
Not advertised to any peer yet
```

The preceding command output shows that the BGP routing table of DeviceA contains two routes destined for 10.22.0.0/16. The route with next hop address 10.1.3.2 is selected because its PrefVal (120) is greater than the PrefVal (0) of the route with next hop address 10.1.2.2. The PrefVal value is sufficient enough to determine the optimal route, and therefore, Device A does not compare other route attributes.

The preceding examples show that PrefVal values can be configured as required to control the traffic forwarding path.

7.3. Local_Pref

BGP prefers the route with the highest Local_Pref during BGP route selection.

The Local_Pref attribute is used to determine the optimal route when traffic leaves an AS. The Local_Pref attribute is available only to IBGP peers and is not advertised to other ASs.

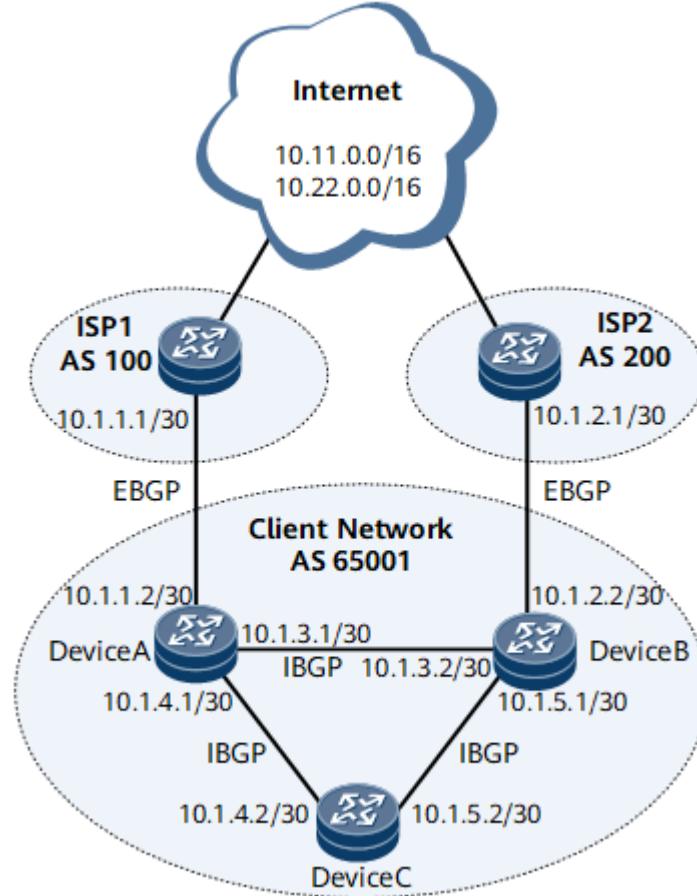
Table 1-138 lists two methods to modify the Local_Pref value.

Table 1-138 Methods to modify the Local_Pref value

Method	Usage Scenario
Run the default local-preference command.	This method sets a default Local_Pref for the routes that the local device advertises to IBGP peers.
Configure an import or export policy and run the apply local-preference command to configure an apply clause for the policy.	This method sets different Local_Pref values for different routes that the local device advertises to IBGP peers. NOTE If a route matches both the default local-preference and apply local-preference commands, the apply local-preference command takes effect.

Figure 1-408 is used as an example to show how the Local_Pref value is used during BGP route selection. In **Figure 1-408**, both ISP1 and ISP2 advertise the routes 10.11.0.0/16 and 10.22.0.0/16 to AS 65001.

Figure 1-408 Local_Pref application networking



Scenario 1: When no Local_Pref value is configured on Device A and Device B, check the BGP routing tables of Device A and Device B.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 192.168.2.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
      Network          NextHop        MED     LocPrf   PrefVal Path/Ogn
      *> 10.1.1.0/30    0.0.0.0       0         0       i
      *>i 10.1.2.0/30   10.1.3.2      0       100      0       i
      *> 10.11.0.0/16   10.1.1.1      100      0       100 10i
      * i               10.1.2.1      100      0       200 10i
      *> 10.22.0.0/16   10.1.1.1      100      0       100 10i
      * i               10.1.2.1      100      0       200 10i
```

The BGP routing table of Device A shows that Device A receives the routes 10.11.0.0/16 and 10.22.0.0/16 from ISP1 and Device B.

```
[~DeviceA] display bgp routing-table 10.11.0.0
```

```
BGP local router ID : 192.168.2.3
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.1.1 (192.168.2.5)
Route Duration: 04h41m03s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 100 10, origin igrp, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.3.2
    10.1.4.2
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.3.2 (192.168.2.2)
Route Duration: 01h42m40s
Relay IP Nexthop: 10.1.3.2
Relay IP Out-Interface: GigabitEthernet1/0/3
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igrp, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for peer type
Not advertised to any peer yet
```

The preceding command output shows that the route learned by DeviceA from ISP1 is selected as the optimal route because it is an EBGP route and the route learned from Device B is an IBGP route. **Table 1-139** shows the route attribute comparison of the routes learned from ISP1 and Device B.

Table 1-139 Route attribute comparison of the routes learned from ISP1 and Device B

Route Attribute	Route Learned from ISP1	Route Learned from Device B	Comparison
PrefVal	0	0	The same.
Local_Pref	-	100	The same. NOTE If a route does not carry Local_Pref, the default value 100 takes effect.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	100 10	200 10	The same length.
Origin	IGP	IGP	The same.
MED	-	-	The same.
Peer type	EBGP	IBGP	The route learned from ISP1 is optimal.

The route selection process on Device B is the same as that on Device A. Then, Device A and Device B advertise the optimal routes to Device C. Check the routing table of Device C.

```
[~DeviceC] display bgp routing-table
Total Number of Routes: 6

BGP Local router ID is 192.168.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
               h - history, i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Network      NextHop     MED     LocPrf   PrefVal Path/Ogn
*>i 10.1.1.0/30    10.1.4.1    0       100      0       i
*>i 10.1.2.0/30    10.1.5.1    0       100      0       i
*>i 10.11.0.0/16   10.1.2.1    100     0       200 10i
* i              10.1.1.1    100     0       100 10i
*>i 10.22.0.0/16   10.1.2.1    100     0       200 10i
* i              10.1.1.1    100     0       100 10i
```

The preceding command output shows that Device C selects the routes advertised by Device B.

Check the reason why the routes learned from Device A are not selected on Device C. The route 10.11.0.0/16 is used as an example.

```
[~DeviceC] display bgp routing-table 10.11.0.0
BGP local router ID : 192.168.2.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.5.1 (192.168.2.2)
Route Duration: 00h12m46s
Relay IP Nexthop: 10.1.5.1
Relay IP Out-Interface: GigabitEthernet1/0/1
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igp, localpref 100, pref-val 0, valid, internal, best, select, active, pre 255
Not advertised to any peer yet

BGP routing table entry information of 10.11.0.0/16:
From: 10.1.4.1 (192.168.2.3)
Route Duration: 00h17m30s
Relay IP Nexthop: 10.1.4.1
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 100 10, origin igp, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for router ID
Not advertised to any peer yet
```

The preceding command output shows that Device C selects the route learned from Device B because the router ID (192.168.2.2) of Device B is smaller than that (192.168.2.3) of Device A. **Table 1-140** shows the route attribute comparison of the routes learned from Device A and Device B.

Table 1-140 Route attribute comparison of the routes learned from Device A and Device B

Route Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
PrefVal	0	0	The same.

Route Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
Local_Pref	100	100	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	100 10	200 10	The same length.
Origin	IGP	IGP	The same.
MED	-	-	The same.
Peer type	IBGP	IBGP	The same.
IGP cost	-	-	The same.
Cluster_List	-	-	The same length.
Router ID	192.168.2.3	192.168.2.2	The route learned from Device B is optimal.

Scenario 2: The administrator of AS 65001 requires that ISP1 be active and ISP2 be backup for the traffic to 10.11.0.0/16 and 10.22.0.0/16.

To meet the preceding requirements, run the **default local-preference** command on DeviceA to increase the Local_Pref values of the routes advertised by DeviceA. This configuration ensures that the routes learned from ISP1 are selected by BGP devices in AS 65001 as the optimal routes to 10.11.0.0/16 and 10.22.0.0/16. Detailed configurations are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
  default local-preference 120          //Set the Local_Pref of the routes to be advertised to 120.  
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device A.

```
[~DeviceA] display bgp routing-table  
BGP Local router ID is 192.168.2.3  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
              h - history, i - internal, s - suppressed, S - Stale  
              Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 4  
  Network      NextHop      MED     LocPrf  PrefVal Path/Ogn  
*> 10.1.1.0/30  0.0.0.0    0        0       i  
*>i 10.1.2.0/30 10.1.3.2   0        100     0       i  
*> 10.11.0.0/16 10.1.1.1   0        0       100 10i  
*> 10.22.0.0/16 10.1.1.1   0        0       100 10i
```

The preceding command output shows that Device A selects the routes learned from ISP1.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table
BGP Local router ID is 192.168.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
  Network      NextHop      MED      LocPrf  PrefVal Path/Ogn
*>i 10.1.1.0/30    10.1.3.1    0       120      0     i
*> 10.1.2.0/30    0.0.0.0    0       0       0     i
*>i 10.11.0.0/16   10.1.1.1    120     0       100 10i
*          10.1.2.1        0       200 10i
*>i 10.22.0.0/16   10.1.1.1    120     0       100 10i
*          10.1.2.1        0       200 10i
```

The preceding command output shows that Device B selects the routes learned from Device A. Device B does not advertise the routes learned from ISP2 to its IBGP peers because those routes are not selected.

Display detailed information about the route 10.11.0.0/16 or 10.22.0.0/16 on Device B. The route 10.11.0.0/16 is used as an example.

```
[~DeviceB] display bgp routing-table 10.11.0.0
BGP local router ID : 192.168.2.2
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.3.1 (192.168.2.3)
Route Duration: 00h22m16s
Relay IP Nexthop: 10.1.3.1
Relay IP Out-Interface: GigabitEthernet1/0/4
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 100 10, origin igrp, localpref 120, pref-val 0, valid, internal, best, select, active, pre 255
Advertised to such 1 peers:
  10.1.2.1
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.2.1 (192.168.2.4)
Route Duration: 00h22m23s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igrp, pref-val 0, valid, external, pre 255, not preferred for Local_Pref
Not advertised to any peer yet
```

The preceding command output shows that the Local_Pref value of the route learned from Device A is greater than that of the route learned from ISP2 and that the route learned from Device A is selected as the optimal route. [Table 1-141](#) shows the route attribute comparison of the routes learned from Device A and ISP2.

Table 1-141 Route attribute comparison of the routes learned from Device A and ISP2

Route Attribute	Route Learned from Device A	Route Learned from ISP2	Comparison
PrefVal	0	0	The same.
Local_Pref	120	-	<p>The route learned from Device A is optimal.</p> <p>NOTE If a route does not carry Local_Pref, the default value 100 takes effect.</p>

Display the routing table of Device C.

```
[~DeviceC] display bgp routing-table
Total Number of Routes: 4

BGP Local router ID is 192.168.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

      Network      NextHop      MED      LocPrf  PrefVal Path/Ogn
*>i 10.1.1.0/30    10.1.4.1    0      120      0      i
*>i 10.1.2.0/30    10.1.5.1    0      100      0      i
*>i 10.11.0.0/16   10.1.1.1    120      0      100 10i
*>i 10.22.0.0/16   10.1.1.1    120      0      100 10i
```

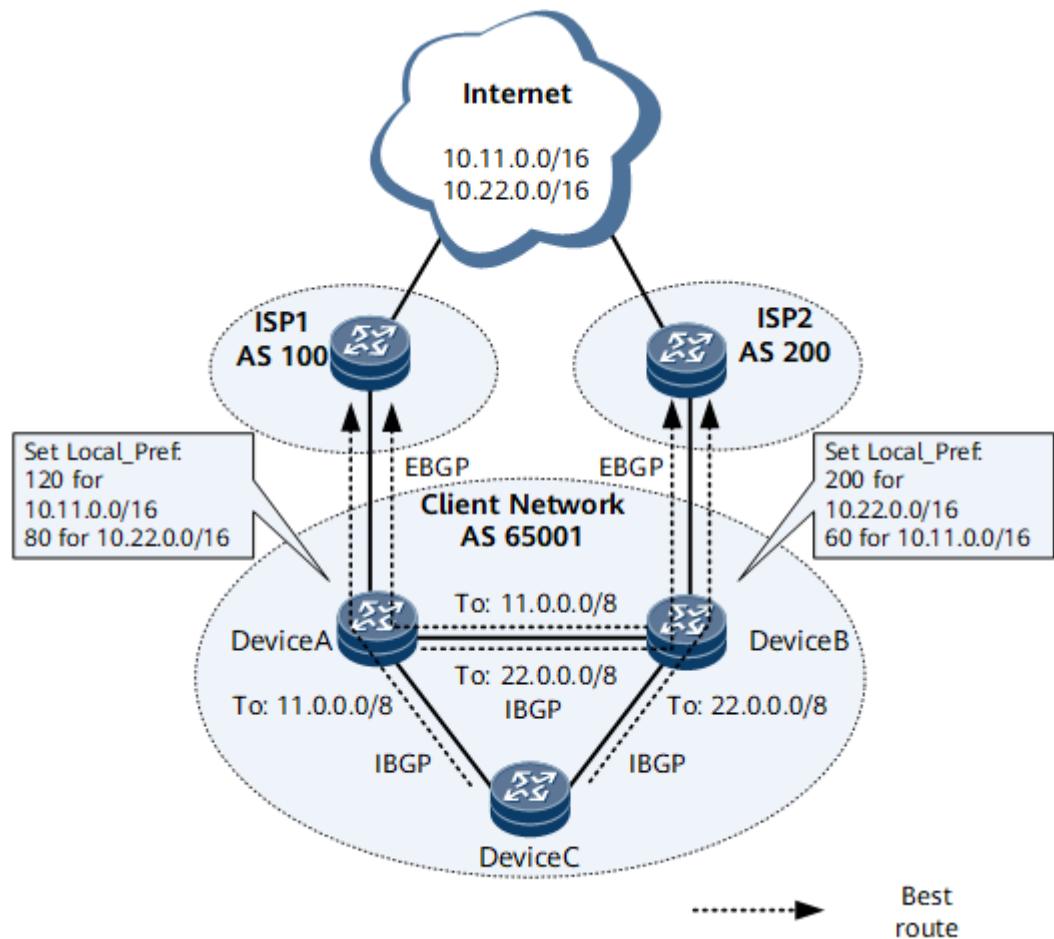
Device C selects the routes advertised by ISP1 because Device B did not advertise the routes learned from ISP2 to Device C.

Scenario 3: The requirements of the administrator of AS 65001 are as follows:

- ISP1 is active and ISP2 is backup for the traffic to 10.11.0.0/16.
- ISP2 is active and ISP1 is backup for the traffic to 10.22.0.0/16.

To meet the preceding requirements, ensure that AS 65001 selects the route 10.11.0.0/16 learned from Device A and the route 10.22.0.0/16 learned from Device B. Detailed configurations are as follows (with [Figure 1-409](#) as the networking):

Figure 1-409 Local_Pref-based BGP route selection networking (1)



In this situation, different Local_Pref values are required for the routes learned from the same ISP. Detailed configurations are as follows:

- Configurations on Device A

```
#  
bgp 65001  
#  
ipv4-family unicast  
peer 10.1.1.1 route-policy rp1 import          //Apply import policy named rp1 to the routes  
learned from 10.1.1.1 and use rp1 to modify the Local_Pref.  
#  
route-policy rp1 permit node 10                  //Define the first node of rp1 and set the Local_Pref  
value of the route 10.11.0.0/16 to 80.  
if-match ip-prefix reducepref  
apply local-preference 80  
#  
route-policy rp1 permit node 20                  //Define the second node of rp1 and set the  
Local_Pref value of the route 10.22.0.0/16 to 120.  
if-match ip-prefix addpref  
apply local-preference 120  
#  
route-policy rp1 permit node 30                  //Define the third node of rp1 and allow rp1 to  
permit all routes.  
#  
ip ip-prefix addpref index 10 permit 10.11.0.0 16 //Configure an IP prefix list to match the route  
10.11.0.0/16.  
ip ip-prefix reducepref index 10 permit 10.22.0.0 16 //Configure an IP prefix list to match the route  
10.22.0.0/16.  
#
```

- Configurations on Device B

```
bgp 65001
#
ipv4-family unicast
peer 10.1.2.1 route-policy rp2 import          //Apply import policy named rp2 to the routes
learned from 10.1.2.1 and use rp2 to modify the Local_Pref.
#
route-policy rp2 permit node 10                  //Define the first node of rp2 and set the Local_Pref
value of the route 10.22.0.0/16 to 200.
if-match ip-prefix addpref
apply local-preference 200
#
route-policy rp2 permit node 20                  //Define the second node of rp2 and set the
Local_Pref value of the route 10.11.0.0/16 to 60.
if-match ip-prefix reducepref
apply local-preference 60
#
route-policy rp2 permit node 30                  //Define the third node of rp2 and allow rp2 to
permit all routes.
#
ip ip-prefix addpref index 10 permit 10.22.0.0 16    //Configure an IP prefix list to match the route
10.22.0.0/16.
ip ip-prefix reducepref index 10 permit 10.11.0.0 16    //Configure an IP prefix list to match the route
10.11.0.0/16.
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device A.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 192.168.2.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 5
      Network        NextHop       MED     LocPrf  PrefVal Path/Ogn
* > 10.1.1.0/30   0.0.0.0      0        0      i
*>i 10.1.2.0/30  10.1.3.2     0      100      0      i
*> 10.11.0.0/16  10.1.1.1     120      0      100 10i
*>i 10.22.0.0/16 10.1.2.1     200      0      200 10i
*           10.1.1.1      80      0      100 10i
```

Display detailed information about the route 10.22.0.0/16 on Device A.

```
[~DeviceA] display bgp routing-table 10.22.0.0
BGP local router ID : 192.168.2.3
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.22.0.0/16:
From: 10.1.3.2 (192.168.2.2)
Route Duration: 00h20m12s
Relay IP Nexthop: 10.1.3.2
Relay IP Out-Interface: GigabitEthernet1/0/3
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igp, localpref 200, pref-val 0, valid, internal, best, select, active, pre 255
Advertised to such 1 peers:
          10.1.1.1
BGP routing table entry information of 10.22.0.0/16:
From: 10.1.1.1 (192.168.2.5)
Route Duration: 00h19m40s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.1.1.1
Qos information : 0x0
```

```
AS-path 100 10, origin igp, localpref 80, pref-val 0, valid, external, pre 255, not preferred for Local_Pref
Not advertised to any peer yet
```

The preceding command output shows that two routes 10.22.0.0/16 are available in the BGP routing table of Device A and that the route with next hop address 10.1.2.1 is selected because its Local_Pref (200) is greater than the Local_Pref (80) of the route with next hop address 10.1.1.1.

Table 1-142 shows the route attribute comparison of the routes learned from ISP1 and Device B.

Table 1-142 Route attribute comparison of the routes learned from ISP1 and Device B.

Route Attribute	Route Learned from ISP1	Route Learned from Device B	Comparison
PrefVal	0	0	The same.
Local_Pref	200	80	The route learned from ISP1 is optimal.

The route with next hop address 10.1.1.1 is not optimal, and therefore, it is not advertised to Device B and Device C. In addition, the BGP routing table of Device A has only one route to the destination address 10.11.0.0/16, and the next hop is 10.1.1.1. This is because the route 10.11.0.0/16 with the next hop being 10.1.2.1 on DeviceB is not the optimal route, and DeviceB does not advertise this route to DeviceA or DeviceC.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table
BGP Local router ID is 192.168.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 5
      Network          NextHop        MED     LocPrf   PrefVal Path/Ogn
*>i 10.1.1.0/30    10.1.3.1      0       100      0       i
*> 10.1.2.0/30    0.0.0.0      0           0       i
*>i 10.11.0.0/16   10.1.1.1      120      0       100 10i
*          10.1.2.1      60       0       200 10i
*> 10.22.0.0/16    10.1.2.1      200      0       200 10i
```

Display detailed information about the route 10.11.0.0/16 on Device B.

```
[~DeviceB] display bgp routing-table 10.11.0.0
BGP local router ID : 192.168.2.2
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.3.1 (192.168.2.3)
Route Duration: 00h40m28s
Relay IP Nexthop: 10.1.3.1
Relay IP Out-Interface: GigabitEthernet1/0/4
Original nexthop: 10.1.1.1
Qos information : 0x0
```

```
AS-path 100 10, origin igp, localpref 120, pref-val 0, valid, internal, best, select, active, pre 255
Advertised to such 1 peers:
  10.1.2.1
    BGP routing table entry information of 10.11.0.0/16:
      From: 10.1.2.1 (192.168.2.4)
      Route Duration: 00h41m00s
      Direct Out-interface: GigabitEthernet1/0/0
      Original nexthop: 10.1.2.1
      Qos information : 0x0
      AS-path 200 10, origin igp, localpref 60, pref-val 0, valid, external, pre 255, not preferred for Local_Pref
      Not advertised to any peer yet
```

The preceding command output shows that two routes 10.11.0.0/16 are available in the BGP routing table of Device B and that the route with next hop address 10.1.1.1 is selected because its Local_Pref (120) is greater than the Local_Pref (60) of the route with next hop address 10.1.2.1. The route with next hop address 10.1.2.1 is not advertised to Device A or Device C because it is not optimal.

Display the routing table of Device C.

```
[~DeviceC] display bgp routing-table
Total Number of Routes: 4

BGP Local router ID is 192.168.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

      Network          NextHop        MED     LocPrf   PrefVal Path/Ogn
*>i 10.1.1.0/30    10.1.4.1      0       100      0     i
*>i 10.1.2.0/30    10.1.5.1      0       100      0     i
*>i 10.11.0.0/16   10.1.1.1      120     0       100 10i
*>i 10.22.0.0/16   10.1.2.1      200     0       200 10i
```

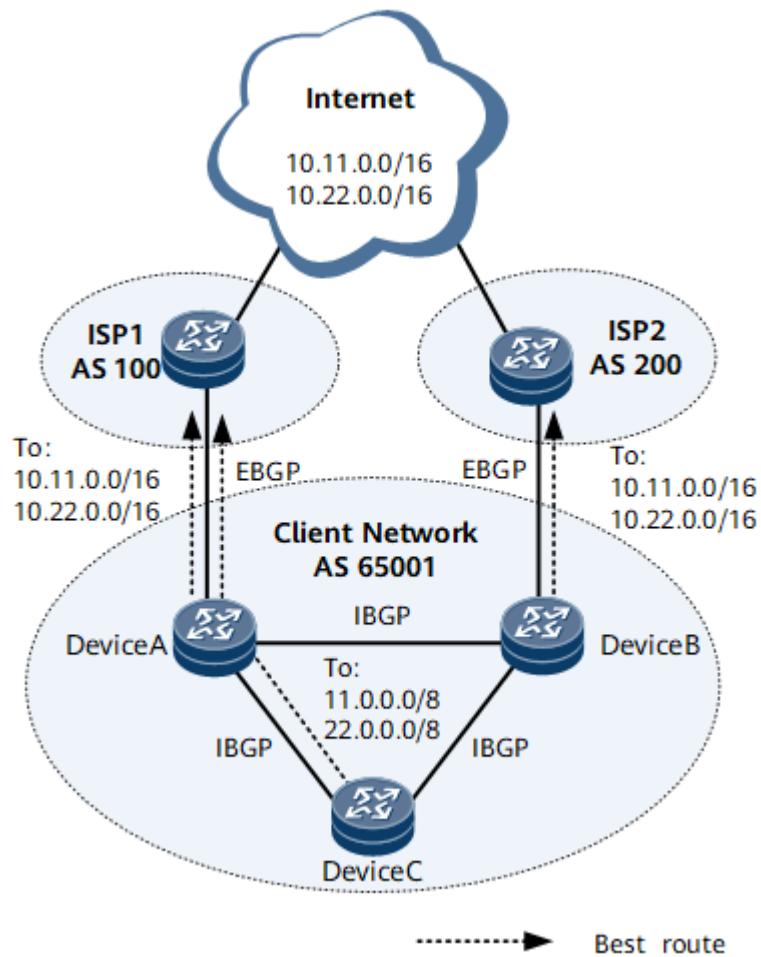
The preceding command output shows that the next hop address of the route 10.11.0.0/16 is 10.1.1.1 and that the next hop address of the route 10.22.0.0/16 is 10.1.2.1.

Scenario 4: The requirements of the administrator of AS 65001 are as follows:

- ISP1 is active and ISP2 is backup for the traffic from Device A and Device C to 10.11.0.0/16 and 10.22.0.0/16.
- ISP2 is active and ISP1 is backup for the traffic from Device B to 10.11.0.0/16 and 10.22.0.0/16.

To meet the preceding requirements, ensure that Device A and Device C select the routes learned from ISP1 as the optimal routes to 10.11.0.0/16 and 10.22.0.0/16. Detailed configurations are as follows (with [Figure 1-410](#) as the networking):

Figure 1-410 Local_Pref-based BGP route selection networking (2)



You can perform either of the following operations:

- Configure an export policy on Device A to modify the Local_Pref of the routes to be advertised to Device C.
- Configure an import policy on Device C to modify the Local_Pref of the routes learned from Device A.

The configurations on Device A are used as an example. Detailed configurations are as follows:

```
bgp 65001
#
ipv4-family unicast
peer 10.1.4.2 route-policy rp2 export          //Apply the export policy rp2 to the routes to be
advertiseds to the peer 10.1.4.2 and modify the Local_Pref of the matched routes.
#
route-policy rp2 permit node 10
if-match ip-prefix addpref
apply local-preference 120
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

```
# Display the routing table of Device A.
```

```
[~DeviceA] display bgp routing-table
```

```
BGP Local router ID is 192.168.2.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
*> 10.1.1.0/30    0.0.0.0    0        0          i
*>i 10.1.2.0/30   10.1.3.2   0        100        0          i
*> 10.11.0.0/16   10.1.1.1   0        0          100 10i
* i             10.1.2.1   100        0          200 10i
*> 10.22.0.0/16   10.1.1.1   0        0          100 10i
* i             10.1.2.1   100        0          200 10i
```

The preceding command output shows that Device A selects the routes learned from ISP1.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table
BGP Local router ID is 192.168.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
*>i 10.1.1.0/30   10.1.3.1   0        100        0          i
*> 10.1.2.0/30   0.0.0.0    0        0          0          i
*> 10.11.0.0/16   10.1.2.1   0        0          200 10i
* i             10.1.1.1   100        0          100 10i
*> 10.22.0.0/16   10.1.2.1   0        0          200 10i
* i             10.1.1.1   100        0          100 10i
```

The preceding command output shows that Device B selects the routes learned from ISP2.

Display the routing table of Device C.

```
[~DeviceC] display bgp routing-table
Total Number of Routes: 6

BGP Local router ID is 192.168.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
*>i 10.1.1.0/30   10.1.4.1   0        120        0          i
*>i 10.1.2.0/30   10.1.5.1   0        100        0          i
*>i 10.11.0.0/16   10.1.1.1   0        120        0          100 10i
* i             10.1.2.1   100        0          200 10i
*>i 10.22.0.0/16   10.1.1.1   0        120        0          100 10i
* i             10.1.2.1   100        0          200 10i
```

The preceding command output shows that Device C selects the routes learned from ISP1.

Display detailed information about the route 10.11.0.0/16 or 10.22.0.0/16 on Device C. The route 10.11.0.0/16 is used as an example.

```
[~DeviceC] display bgp routing-table 10.11.0.0
```

```
BGP local router ID : 192.168.2.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.11.0.0/16:
From: 10.1.4.1 (192.168.2.3)
Route Duration: 00h06m26s
Relay IP Nexthop: 10.1.4.1
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 100 10, origin igr, localpref 120, pref-val 0, valid, internal, best, select, active, pre 255
Not advertised to any peer yet

BGP routing table entry information of 10.11.0.0/16:
From: 10.1.5.1 (192.168.2.2)
Route Duration: 00h08m05s
Relay IP Nexthop: 10.1.5.1
Relay IP Out-Interface: GigabitEthernet1/0/1
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igr, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for Local_Pref
Not advertised to any peer yet
```

The preceding command output shows that Device C selects the routes learned from ISP1 because the Local_Pref of the routes learned from ISP1 is greater than that of the route learned from ISP2.

The preceding examples show that the modification of the Local_Pref values affects not only BGP route advertisement but also BGP route selection with an AS. We can configure Local_Pref values as required to control the forwarding path of the traffic that leaves an AS.

7.4. Route Type

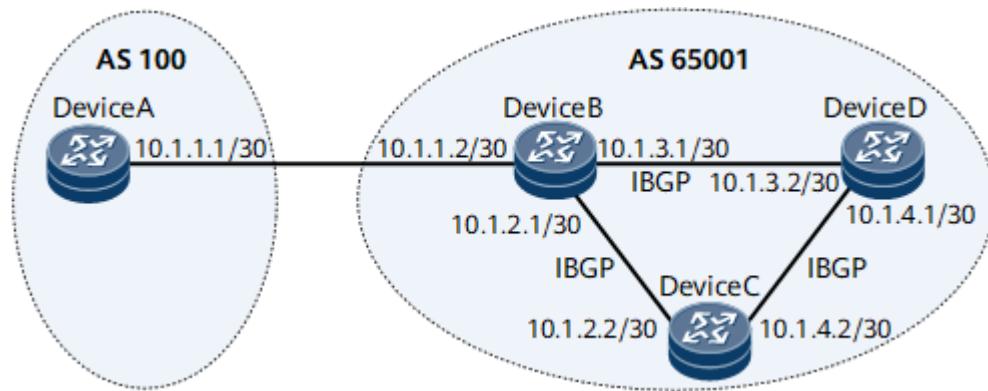
BGP prefers locally imported routes to the routes learned from peers during BGP route selection.

BGP routes can be locally imported or learned from peers. The locally imported routes take precedence over the routes learned from peers during BGP route selection. It is unusual for locally imported routes and the routes learned from peers to carry the same destination IP address and coexist in the routing table. Generally, locally imported routes can be the routes imported using the **network** or **import-route** command and the automatically and manually summary routes. Precedences of these routes are described as follows:

1. Summary routes take precedence over non-summary routes.
2. Summary routes that are manually generated using the **aggregate** command take precedence over summary routes that are automatically generated based on the **summary automatic** command settings.
3. The routes imported using the **network** command take precedence over the routes imported using the **import-route** command.

In **Figure 1-411**, Device A and Device B are EBGP peers, and Device B, Device C, and Device D are IBGP peers.

Figure 1-411 Networking



The configurations on Device C are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
network 10.1.2.0 255.255.255.252          //Advertise the route 10.1.2.0/30.  
network 10.1.4.0 255.255.255.252          //Advertise the route 10.1.4.0/30.  
import-route direct                         //Import direct routes.  
#
```

The configurations on Device D are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
network 10.1.3.0 255.255.255.252          //Advertise the route 10.1.3.0/30.  
network 10.1.4.0 255.255.255.252          //Advertise the route 10.1.4.0/30.  
import-route direct                         //Import direct routes.  
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device D.

```
[~DeviceD] display bgp routing-table  
BGP Local router ID is 10.1.3.2  
  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
              h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 10  
      Network      NextHop      MED      LocPrf      PrefVal Path/Ogn  
*>i 10.1.2.0/30    10.1.4.2      0      100      0      i  
*> 10.1.3.0/30    0.0.0.0      0      0      0      i  
*       0.0.0.0      0      0      ?  
*> 10.1.3.2/32    0.0.0.0      0      0      ?  
*> 10.1.4.0/30    0.0.0.0      0      0      i  
*       0.0.0.0      0      0      ?  
i       10.1.4.2      0      100      0      ?  
*> 10.1.4.1/32    0.0.0.0      0      0      ?  
*> 127.0.0.0      0.0.0.0      0      0      ?  
*> 127.0.0.1/32   0.0.0.0      0      0      ?
```

The preceding command output shows that three routes 10.1.4.0/30 are available in the routing table. The route with the next hop address 10.1.4.2 is learned from a peer (Device C). Therefore, BGP first excludes this route from route selection.

```
[~DeviceD] display bgp routing-table 10.1.4.0 30
BGP local router ID : 10.1.3.2

Local AS number : 65001
Paths: 3 available, 1 best, 1 select
BGP routing table entry information of 10.1.4.0/30:
Network route.
From: 0.0.0.0 (0.0.0.0)
Route Duration: 00h03m51s
Direct Out-interface: GigabitEthernet1/0/4
Original nexthop: 10.1.4.1
Qos information : 0x0
AS-path Nil, origin igrp, MED 0, pref-val 0, valid, local, best, select, pre 0
Advertised to such 2 peers:
    10.1.3.1
    10.1.4.2
BGP routing table entry information of 10.1.4.0/30:
Imported route.
From: 0.0.0.0 (0.0.0.0)
Route Duration: 00h04m10s
Direct Out-interface: GigabitEthernet1/0/4
Original nexthop: 10.1.4.1
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, pref-val 0, valid, local, pre 0, not preferred for route type
Not advertised to any peer yet

BGP routing table entry information of 10.1.4.0/30:
From: 10.1.4.2 (10.1.2.2)
Route Duration: 00h02m24s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface: GigabitEthernet1/0/4
Original nexthop: 10.1.4.2
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, internal, pre 255
Not advertised to any peer yet
```

The preceding command output shows that the route imported using the **network** command is selected as the optimal route.

The configurations on Device B are as follows:

```
bgp 65001
#
ipv4-family unicast
summary automatic
aggregate 10.0.0.0 255.0.0.0
import-route direct
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table
BGP Local router ID is 10.1.1.2

Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 14
      Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
```

```
*> 10.0.0.0      127.0.0.1          0    ?
*           127.0.0.1          0    ?
s> 10.1.1.0/30  0.0.0.0      0    0    ?
*> 10.1.1.2/32  0.0.0.0      0    0    ?
s> 10.1.2.0/30  0.0.0.0      0    0    ?
i           10.1.2.2      0    100   0    i
*> 10.1.2.1/32  0.0.0.0      0    0    ?
s> 10.1.3.0/30  0.0.0.0      0    0    ?
i           10.1.3.2      0    100   0    i
*> 10.1.3.1/32  0.0.0.0      0    0    ?
*>i 10.1.4.0/30 10.1.3.2      0    100   0    i
*i           10.1.2.2      0    100   0    ?
*> 127.0.0.0    0.0.0.0      0    0    ?
*> 127.0.0.1/32 0.0.0.0      0    0    ?
```

The preceding command output shows that two summary routes 10.0.0.0 are available in the routing table.

```
[~DeviceB] display bgp routing-table 10.0.0.0
BGP local router ID : 10.1.1.2

Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.0.0.0/8:
Aggregated route.
Route Duration: 00h17m04s
Direct Out-interface: NULL0
Original nexthop: 127.0.0.1
Qos information : 0x0
AS-path Nil, origin incomplete, pref-val 0, valid, local, best, select, active, pre 255
Aggregator: AS 65001, Aggregator ID 10.1.1.2
Advertised to such 3 peers:
  10.1.1.1
  10.1.3.2
  10.1.2.2
BGP routing table entry information of 10.0.0.0/8:
Summary automatic route
Route Duration: 00h17m04s
Direct Out-interface: NULL0
Original nexthop: 127.0.0.1
Qos information : 0x0
AS-path Nil, origin incomplete, pref-val 0, valid, local, pre 255, not preferred for route type
Aggregator: AS 65001, Aggregator ID 10.1.1.2
Not advertised to any peer yet
```

The preceding command output shows that the route generated using the **aggregate** command is selected as the optimal route.

7.5. AIGP

BGP prefers the route with the smallest AIGP value during BGP route selection.

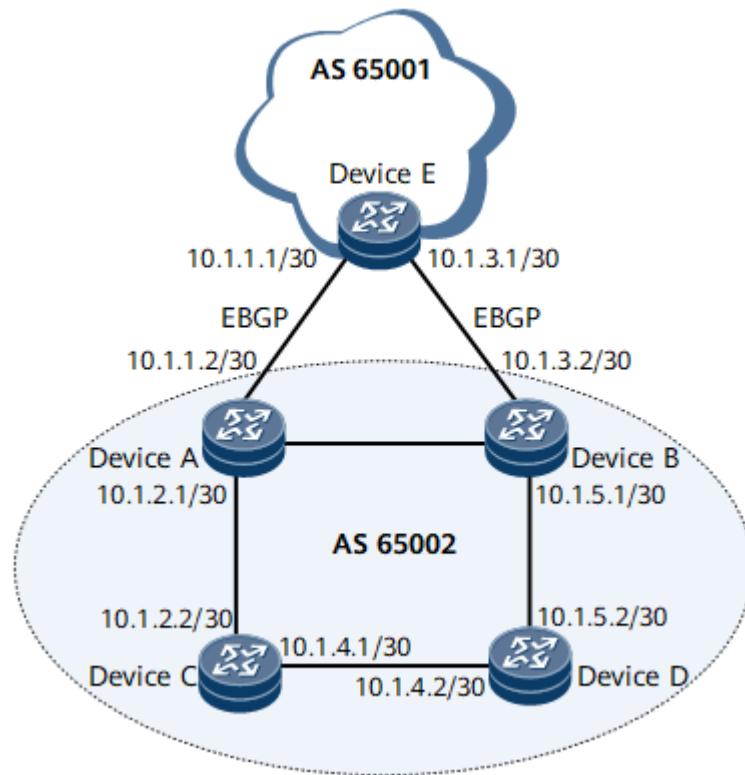
The Accumulated Interior Gateway Protocol Metric (AIGP) attribute is an optional non-transitive Border Gateway Protocol (BGP) path attribute. After the AIGP attribute is configured in an AIGP administrative domain, BGP selects paths based on costs in the same manner as an IGP, and all devices in the domain forward data along the optimal routes. During BGP route selection, the AIGP attribute is used as follows:

- The priority of a route that carries the AIGP attribute is higher than the priority of a route that does not carry the AIGP attribute.
- If routes carry the AIGP attribute, the device selects the route whose AIGP value plus the cost of the IGP route to which the route recurses is smaller.

The AIGP attribute can be added to routes only through route-policies. You can configure an apply clause for a route-policy using the **apply aigp { [+ | -] cost | inherit-cost }** command to modify the AIGP value during route import, acceptance, or advertisement. If no AIGP value is configured, the IGP routes imported by BGP do not carry the AIGP attribute.

Figure 1-412 is used as an example to show how the AIGP attribute is used during BGP route selection. In **Figure 1-412**, OSPF runs in AS 65002, and an EBGP peer relationship is established between DeviceA and DeviceE, and between DeviceB and DeviceE. BGP is configured on DeviceA and DeviceB to import OSPF routes in AS 65002 and advertise the routes to AS 65001.

Figure 1-412 AIGP application networking



Run the **display bgp routing-table [ip-address]** command on Device E to check the configurations. The route 10.1.4.0/30 is used in this example.

Display the routing table of Device E.

```
[~DeviceE] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
      Network          NextHop        MED     LocPrf  PrefVal Path/Ogn
    *> 10.1.2.0/30    10.1.1.2       0         0   65002?
    *       10.1.3.2       3         0   65002?
    *> 10.1.4.0/30    10.1.1.2       2         0   65002?
    *       10.1.3.2       2         0   65002?
```

```
*> 10.1.5.0/30      10.1.3.2      0          0      65002?
*           10.1.1.2      3          0      65002?
[~DeviceE] display bgp routing-table 10.1.4.0
BGP local router ID : 10.1.1.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.1.2 (10.1.1.2)
Route Duration: 00h02m29s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.1.2
    10.1.3.2
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.3.2 (10.1.5.1)
Route Duration: 00h03m58s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, pre 255, not preferred for router ID
Not advertised to any peer yet
```

The command output when the AIGP attribute is not configured shows that Device E selects the route learned from Device A after comparing router IDs. To change the route selection result on Device E, you can configure the AIGP attribute.

Configurations on Device A:

```
#  
bgp 65002  
#  
ipv4-family unicast  
import-route ospf 1 route-policy aigp_policy //Apply route-policy named aigp_policy to locally  
imported OSPF routes and use aigp_policy to modify the AIGP value.  
peer 10.1.1.1 aigp //Enable AIGP on the local device and the peer 10.1.1.1.  
#  
route-policy aigp_policy permit node 10 //Define the first node of aigp_policy and set the AIGP  
value of the route 10.1.4.0/30 to 10.  
if-match ip-prefix prefix1  
apply aigp 10  
#  
route-policy aigp_policy permit node 20 //Define the second node of aigp_policy and allow  
aigp_policy to permit all routes.  
#  
ip ip-prefix prefix1 index 10 permit 10.1.4.0 30 //Configure IP prefix list named prefix1 to match the  
route 10.1.4.0/30.  
#
```

Configurations on Device B:

```
bgp 65002  
peer 10.1.3.1 as-number 65001  
#  
ipv4-family unicast  
import-route ospf 1 route-policy aigp_policy1 //Apply route-policy named aigp_policy1 to locally  
imported OSPF routes and use aigp_policy1 to modify the AIGP value.  
peer 10.1.3.1 aigp //Enable AIGP on the local device and the peer 10.1.3.1.  
#  
route-policy aigp_policy1 permit node 10 //Define the first node of aigp_policy1 and set the AIGP  
value of the route 10.1.4.0/30 to 5.  
if-match ip-prefix prefix2  
apply aigp 5  
#  
route-policy aigp_policy1 permit node 20 //Define the second node of aigp_policy1 and allow  
aigp_policy1 to permit all routes.
```

```
#  
ip ip-prefix prefix2 index 10 permit 10.1.4.0 30 //Configure IP prefix list named prefix2 to match the  
route 10.1.4.0/30.  
#
```

Configurations on Device E:

```
#  
bgp 65001  
#  
ipv4-family unicast  
peer 10.1.1.2 aigp //Enable AIGP on the local device and the peer 10.1.1.2.  
peer 10.1.3.2 aigp //Enable AIGP on the local device and the peer 10.1.3.2.  
#
```

Run the **display bgp routing-table [ip-address]** command on Device E to check the configurations.

Display the routing table of Device E.

```
[~DeviceE] display bgp routing-table  
BGP Local router ID is 10.1.1.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 6  
Network NextHop MED LocPrf PrefVal Path/Ogn  
*> 10.1.2.0/30 10.1.1.2 0 0 65002?  
* 10.1.3.2 3 0 65002?  
*> 10.1.4.0/30 10.1.3.2 2 0 65002?  
* 10.1.1.2 2 0 65002?  
*> 10.1.5.0/30 10.1.3.2 0 0 65002?  
* 10.1.1.2 3 0 65002?  
[~DeviceE] display bgp routing-table 10.1.4.0  
BGP local router ID : 10.1.1.1  
Local AS number : 65001  
Paths: 2 available, 1 best, 1 select  
BGP routing table entry information of 10.1.4.0/30:  
From: 10.1.3.2 (10.1.5.1)  
Route Duration: 00h00m14s  
Direct Out-interface: GigabitEthernet1/0/0  
Original nexthop: 10.1.3.2  
Qos information : 0x0  
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, best, select, active, pre 255, AIGP 5  
Advertised to such 2 peers:  
10.1.1.2  
10.1.3.2  
BGP routing table entry information of 10.1.4.0/30:  
From: 10.1.1.2 (10.1.1.2)  
Route Duration: 01h01m15s  
Direct Out-interface: GigabitEthernet1/0/1  
Original nexthop: 10.1.1.2  
Qos information : 0x0  
AS-path 65002, origin incomplete, MED 2, pref-val 0, valid, external, pre 255, AIGP 10, not preferred for AIGP  
Not advertised to any peer yet
```

The preceding command output shows that Device E selects the route 10.1.4.0/30 learned from Device B because its AIGP value is smaller.

Table 1-143 shows the attribute comparison of the routes learned from Device A and Device B.

Table 1-143 Attribute comparison of the routes learned from Device A and Device B.

Route Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
PrefVal	0	0	The same.
Local_Pref	-	-	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	10	5	The different.

?6. AS_Path

BGP prefers the route with the shortest AS_Path length (the number of included ASs) during BGP route selection.

Four types of AS_Path attributes are available: AS_Sequence, AS_Set, AS_Confederation_Sequence, and AS_Confederation_Set.

- AS_Sequence: records in reverse order all the ASs through which a route passes from the local device to the destination.
- AS_Set: records in random order all the ASs through which a route passes from the local device to the destination. The AS_Set attribute is used in route summarization scenarios.
- AS_Confederation_Sequence: records in reverse order all the sub-ASs within a BGP confederation through which a route passes from the local device to the destination.
- AS_Confederation_Set: records in random order all the sub-ASs within a BGP confederation through which a route passes from the local device to the destination.

Table 1-144 describes the AS_Path-based route selection rules.

Table 1-144 AS_Path-based route selection rules

Item	Description
AS_Set	<p>BGP takes an AS_Set as one AS number even if the AS_Set contains multiple AS numbers.</p> <p>When the aggregate (BGP) command is run to manually generate a summary route, if as-set is specified in the command, an AS_Set will be carried in the summary route. Otherwise, no AS_Set will be carried. The information in the AS_Set is as follows:</p> <ul style="list-style-type: none"> • If the routes used in the summarization carry the same AS_Sequence, this AS_Sequence is carried in the summarized route, and the AS_Set of the summarized route is null. • If the routes used in the summarization carry different AS_Sequences, all the AS numbers carried in the AS_Sequences are included in the AS_Set of the summarized route.
AS_Confed_Sequence and AS_Confed_Set	BGP ignores AS_Confed_Sequence and AS_Confed_Set when calculating the AS_Path length.
bestroute as-path-ignore	After the command is run, BGP does not compare the AS_Path lists carried in candidate routes during route selection.
apply as-path	<p>The command can be run to configure an apply clause for a route-policy so that the ASs in the AS_Path of the route that matches the route-policy are cleared or replaced, or new ASs are added.</p> <p>NOTE The configuration of the apply as-path command may change the traffic forwarding path, or cause routing loops or route selection errors. Therefore, exercise caution when configuring the command.</p>
peer public-as-only	After the command is run, BGP removes all the private AS numbers (if any) from the AS_Path attribute in each Update message to be sent. The private AS number ranges from 64512 to 65534 and from 4200000000 to 4294967294 (or from 64086.59904 to 65535.65534).
peer public-as-only import	After the command is run, BGP removes all the private AS numbers (if any) from the AS_Path attribute in each received Update message. The private AS number ranges from 64512 to 65534 and from 4200000000 to 4294967294 (or from 64086.59904 to 65535.65534).

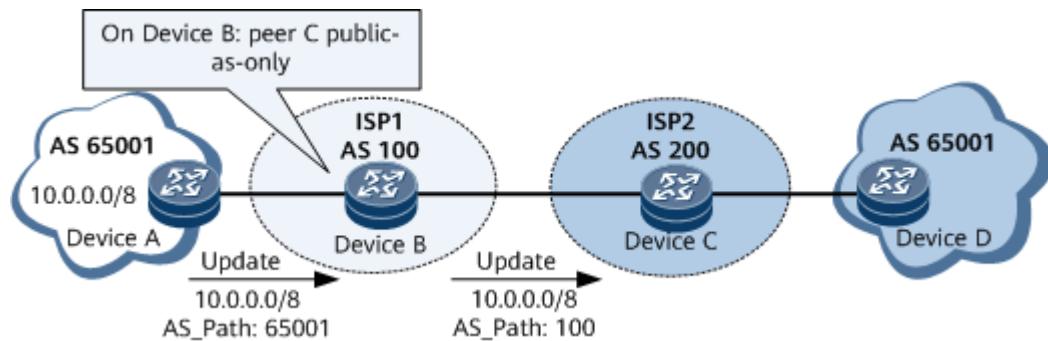
Item	Description
peer fake-as	<p>After the command is run, BGP can use a fake AS number to set up a BGP peer relationship.</p> <p>If the local device uses the actual AS number to establish an EBGP peer relationship with a remote device, the actual AS number is carried in the AS_Path of the route sent to the remote device. If the local device uses the fake AS number to establish the EBGP peer relationship, the fake AS number is carried in the AS_Path of the route sent to the remote device.</p>
peer substitute-as	<p>After the command is run, if the AS_Path attribute in the route that a PE will send to its connected CE through a BGP peer relationship contains an AS number the same as that of the CE, the PE replaces the AS number in the AS_Path attribute with its local AS number before advertising this route.</p> <p>NOTE The peer substitute-as command applies only to PEs in BGP MPLS IP/VPN scenarios and may cause routing loops if it is improperly configured. Therefore, exercise caution when using the command.</p>

During BGP route selection, BGP compares the AS_Path length by calculating the number of ASs included in the AS_Sequence if AS_Sequence is carried in a route. If both AS_Sequence and AS_Set are carried in the route, BGP considers the AS_Path length to be the number of ASs included in the AS_Sequence plus 1. The following describes some common commands in detail by using examples.

Deleting Private AS Numbers

As public AS resources are limited, carriers generally use private AS numbers when deploying VPNs. Private AS numbers, however, must not be advertised to the Internet because they may cause routing loops. In [Figure 1-413](#), both ISP1 and ISP2 use 65001 as a private AS number.

[Figure 1-413](#) Networking in which a private AS needs to be deleted



In [Figure 1-413](#), DeviceA advertises the route 10.0.0.0/8 to DeviceD through ISP1 and ISP2. After receiving this route, DeviceD checks its AS_Path attribute. This

AS_Path attribute carries AS 65001, which is the same as the AS number of DeviceD. As a result, DeviceD discards this route.

To address this problem, run the **peer public-as-only** command on DeviceB so that DeviceB in ISP1 deletes AS 65001 before adding AS 100 (public AS) to the AS_Path attribute and forwarding the route to DeviceC in ISP2.

Before using the **peer public-as-only** command, note the following restrictions: BGP does not remove private AS numbers in the following scenarios:

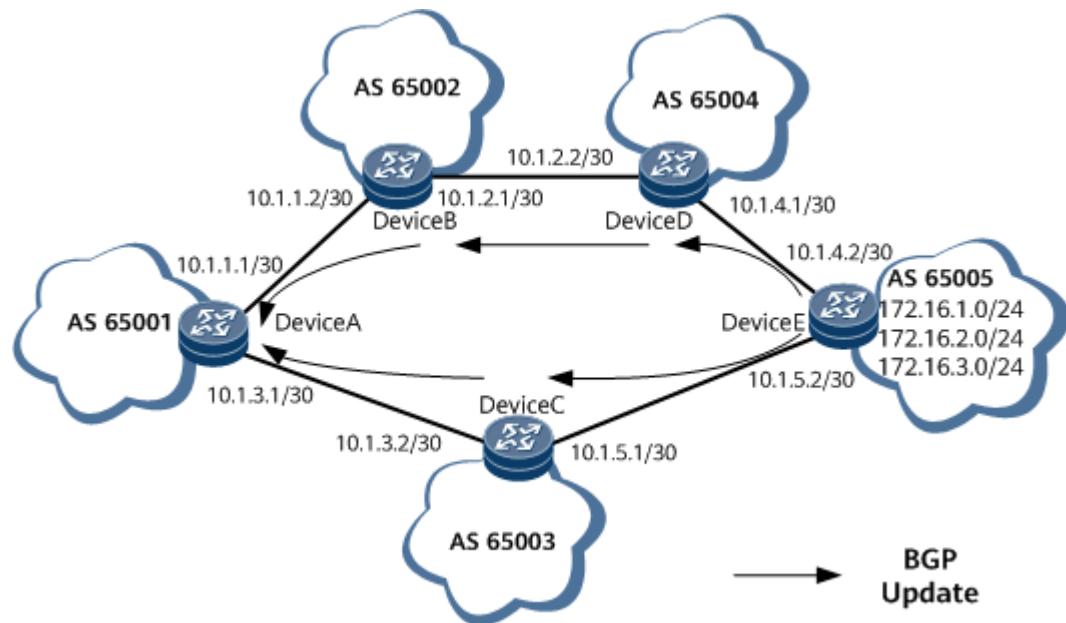
- The AS_Path of a route carries the AS number of the remote peer. In this case, deleting private AS numbers may lead to a routing loop.
- The AS_Path carries both public and private AS numbers, which indicates that the route has passed through the public network. In this case, deleting private AS numbers may lead to incorrect traffic forwarding.

The preceding limitations also apply to confederation scenarios.

Adding AS Numbers

In [Figure 1-414](#), AS 65005 imports three routes and advertises them to AS 65001 through two paths.

Figure 1-414 Networking in which new AS numbers are added to the AS_Path



Run the **display bgp routing-table [ip-address]** command to verify the configuration.

Display the routing table of DeviceA.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found
Total Number of Routes: 6
```

Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
*> 172.16.1.0/24	10.1.3.2		0	65003	65005?
*	10.1.1.2		0	65002	65004 65005?
*> 172.16.2.0/24	10.1.3.2		0	65003	65005?
*	10.1.1.2		0	65002	65004 65005?
*> 172.16.3.0/24	10.1.3.2		0	65003	65005?
*	10.1.1.2		0	65002	65004 65005?
[~DeviceA] display bgp routing-table 172.16.1.0					
BGP local router ID : 10.1.1.1					
Local AS number : 65001					
Paths: 2 available, 1 best, 1 select					
BGP routing table entry information of 172.16.1.0/24:					
From: 10.1.3.2 (10.1.5.1)					
Route Duration: 00h00m56s					
Direct Out-interface: GigabitEthernet1/0/0					
Original nexthop: 10.1.3.2					
Qos information : 0x0					
AS-path 65003 65005 , origin incomplete, pref-val 0, valid, external, best , select, active, pre 255					
Advertised to such 2 peers:					
10.1.3.2					
10.1.1.2					
BGP routing table entry information of 172.16.1.0/24:					
From: 10.1.1.2 (10.1.1.2)					
Route Duration: 00h34m43s					
Direct Out-interface: GigabitEthernet2/0/0					
Original nexthop: 10.1.1.2					
Qos information : 0x0					
AS-path 65002 65004 65005 , origin incomplete, pref-val 0, valid, external, pre 255, not preferred for AS-Path					
Not advertised to any peer yet					

The preceding command output shows that DeviceA selects the route learned from DeviceC because this route has a shorter AS_Path length than that learned from DeviceB. To enable DeviceA to select the route learned from DeviceB, configure DeviceB to reduce the AS_Path length of the route or configure DeviceC to increase the AS_Path length of the route. In the following example, DeviceC is configured to increase the AS_Path length of the route. The detailed configurations on DeviceC are as follows:

```
#  
bgp 65003  
#  
ipv4-family unicast  
undo synchronization  
peer 10.1.3.1 route-policy add_asn export      //Apply export policy named add_asn to the routes to  
be advertised to BGP peer 10.1.3.1.  
#  
route-policy add_asn permit node 10                //Define the first node of add_asn.  
if-match ip-prefix prefix1                         //Configure IP prefix list named prefix1.  
apply as-path 65003 65003 65003 additive        //Add 65003, 65003, 65003 to the AS_Path of the  
route that matches the IP prefix list prefix1.  
#  
route-policy add_asn permit node 20                //Define the second node of add_asn to permit all other  
routes.  
#  
ip ip-prefix prefix1 index 10 permit 172.16.1.0 24 //Define the first index of prefix1 to match the route  
172.16.1.0/24.  
ip ip-prefix prefix1 index 20 permit 172.16.2.0 24 //Define the second index of prefix1 to match the  
route 172.16.2.0/24.  
ip ip-prefix prefix1 index 30 permit 172.16.3.0 24 //Define the third index of prefix1 to match the route  
172.16.3.0/24.
```

Run the **display bgp routing-table [ip-address]** command to verify the configuration.

Display the routing table of DeviceA.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
               h - history, i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
               RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
      Network          NextHop        MED     LocPrf   PrefVal Path/Ogn
*> 172.16.1.0/24    10.1.1.2          0  65002 65004 65005?
*   10.1.3.2          0  65003 65003 65003 65003 65005?
*> 172.16.2.0/24    10.1.1.2          0  65002 65004 65005?
*   10.1.3.2          0  65003 65003 65003 65003 65005?
*> 172.16.3.0/24    10.1.1.2          0  65002 65004 65005?
*   10.1.3.2          0  65003 65003 65003 65003 65005?

[~DeviceA] display bgp routing-table 172.16.1.0
BGP local router ID : 10.1.1.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.1.2 (10.1.1.2)
Route Duration: 00h33m30s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 65002 65004 65005, origin incomplete, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
  10.1.3.2
  10.1.1.2
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.3.2 (10.1.5.1)
Route Duration: 00h02m12s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 65003 65003 65003 65003 65005, origin incomplete, pref-val 0, valid, external, pre 255, not preferred for AS-Path
Not advertised to any peer yet
```

The preceding command output shows that the AS_Path length of the route learned from DeviceB is shorter than that of the route learned from DeviceC and that the route learned from DeviceB is selected as the optimal route. **Table 1-145** shows the attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC.

Table 1-145 Attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC

Route Attribute	Route Learned from DeviceB	Route Learned from DeviceC	Comparison
PrefVal	0	0	The same.
Local_Pref	-	-	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	65002 65004 65005	65003 65003 65003 65003 65005	The route learned from DeviceB is optimal.

AS numbers can be added to the AS_Path as required. However, if an AS number is added to the AS_Path of a route, the route cannot be received by devices in this AS. Therefore, the local AS number is added in most cases. For example in [Figure 1-414](#), if DeviceC adds AS 65001 to the AS_Path of a route before advertising the route to DeviceA, DeviceA will discard the route upon receipt because the route carries DeviceA's AS number.

Replacing AS Numbers

If **overwrite** is specified in the **apply as-path** command, the AS numbers in the AS_Path attribute can be replaced. AS number replacement can be flexibly applied to the following scenarios:

- Shield the actual path information.
- Prevent a route from being discarded by replacing the AS_Path attribute of the route with a shorter one if the **as-path-limit** command is run on the device that receives this route.
- Reduce the AS_Path length.

AS number replacement can also be used for the purpose of load balancing. Generally, BGP requires that the AS_Path attributes of the routes be the same so that load balancing can be implemented. To meet load balancing requirements, AS numbers can be replaced. For example in [Figure 1-414](#), the **apply as-path 65002 65004 65005 overwrite** command can be run on DeviceA to replace the AS_Path of the route learned from DeviceC so that the route has the same AS_Path as that of the route learned from DeviceB, and the two routes are used to load-balance traffic. Detailed configurations on DeviceA are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
undo synchronization  
peer 10.1.3.2 route-policy replace_asn import //Apply export policy named replace_asn to routes to  
be advertised to BGP peer 10.1.3.2.  
#  
route-policy replace_asn permit node 10 //Define the first node of replace_asn.  
if-match as-path-filter filter1 //Configure AS_Path filter named filter1.  
apply as-path 65002 65004 65005 overwrite //Replace the AS_Path of the route that matches  
filter1 with 65002, 65004, 65005.  
#  
route-policy replace_asn permit node 20 //Define the second node of replace_asn to permit all  
other routes.  
#  
ip as-path-filter filter1 permit ^65003 //Define AS_Path filter named filter1 to match all the  
routes learned from AS 65003.  
#
```

Run the **display bgp routing-table [ip-address]** command to verify the configuration.

Display the routing table of DeviceA.

```
[~DeviceA] display bgp routing-table  
BGP Local router ID is 10.1.1.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 6						
	Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
*>	172.16.1.0/24	10.1.1.2		0	65002 65004 65005?	
*		10.1.3.2		0	65002 65004 65005?	
*>	172.16.2.0/24	10.1.1.2		0	65002 65004 65005?	
*		10.1.3.2		0	65002 65004 65005?	
*>	172.16.3.0/24	10.1.1.2		0	65002 65004 65005?	
*		10.1.3.2		0	65002 65004 65005?	

The preceding command output shows that the AS_Path of the route received from AS 65003 has been replaced. In this case, the routes sent from AS 65002 and AS 65003 have the same AS_Path, which meets the load balancing conditions. Run the **maximum load-balancing 2** command on DeviceA to set the maximum number of routes for load balancing to 2. Then, check the detailed BGP route information. The route 172.16.1.0/24 is used in the following example:

```
[~DeviceA] display bgp routing-table 172.16.1.0
BGP local router ID : 10.1.1.1
Local AS number : 65001
Paths: 2 available, 1 best, 2 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.1.2 (10.1.1.2)
Route Duration: 19h57m51s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 65002 65004 65005, origin incomplete, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.1.2
    10.1.3.2
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.3.2 (10.1.5.1)
Route Duration: 00h10m21s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path 65002 65004 65005, origin incomplete, pref-val 0, valid, external, select, active, pre 255, not preferred for router ID
Not advertised to any peer yet
```

The preceding command output shows that the route learned from DeviceB is optimal and is used by BGP along with the route learned from DeviceC (not optimal) for load balancing. Check the information about the route 172.16.1.0/24 in the IP routing table.

```
[~DeviceA] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _Public_
Destinations : 9      Routes : 12
-----
Destination/Mask Proto Pre Cost Flags NextHop      Interface
10.1.1.0/30 Direct 0   0       D  10.1.1.1    GigabitEthernet2/0/0
10.1.1.1/32 Direct 0   0       D  127.0.0.1   GigabitEthernet2/0/0
10.1.3.0/30 Direct 0   0       D  10.1.3.1    GigabitEthernet1/0/0
10.1.3.1/32 Direct 0   0       D  127.0.0.1   GigabitEthernet1/0/0
127.0.0.0/8 Direct 0   0       D  127.0.0.1   InLoopBack0
127.0.0.1/32 Direct 0   0       D  127.0.0.1   InLoopBack0
172.16.1.0/24 EBGP  255  0       D  10.1.1.2    GigabitEthernet2/0/0
                  EBGP  255  0       D  10.1.3.2    GigabitEthernet1/0/0
172.16.2.0/24 EBGP  255  0       D  10.1.1.2    GigabitEthernet2/0/0
                  EBGP  255  0       D  10.1.3.2    GigabitEthernet1/0/0
172.16.3.0/24 EBGP  255  0       D  10.1.1.2    GigabitEthernet2/0/0
                  EBGP  255  0       D  10.1.3.2    GigabitEthernet1/0/0
```

The preceding command output shows that BGP has delivered the two routes with the same route prefix to the IP routing table for load balancing.

Clearing the AS_Path

If **none overwrite** is specified in the **apply as-path** command, the device can clear the AS_Path attribute to shield the actual path information as well as shortening the AS_Path length. If the AS_Path attribute is empty, BGP considers its length as 0 during route selection.

?7. Origin

The Origin attribute indicates how routes become BGP routes.

Three types of Origin attributes are available:

- **IGP**: indicates that routes are added to the BGP routing table using the **network** command. **IGP** has the highest priority.
- **EGP**: indicates that routes are learned through the EGP protocol. **EGP** has the second highest priority.

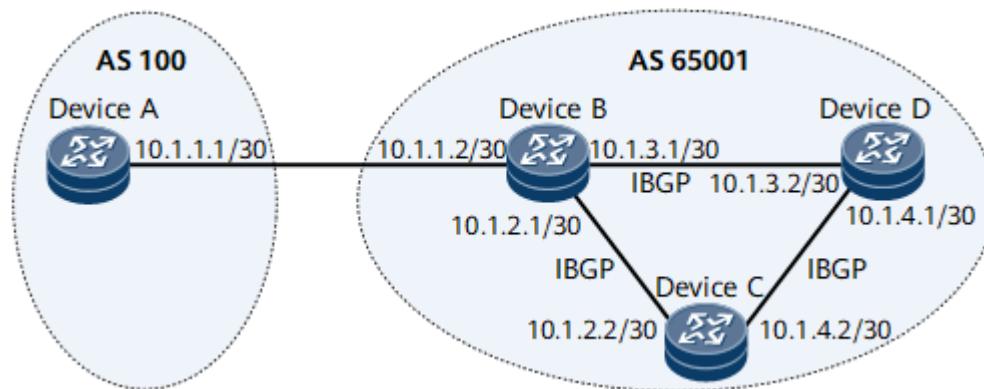
NOTE

The device can receive and send BGP routes with EGP as the Origin. However, devices do not support EGP; therefore, to set the Origin of routes to EGP, you need to run the **apply origin { egp { as-number-plain | as-number-dot } | igr | incomplete }** command to apply a route-policy.

- **Incomplete**: This attribute type has the lowest priority. If a route is added to the BGP routing table using the **import-route** command, the Origin attribute of the route is Incomplete.

The routes with the Origin attribute being IGP have a higher priority than those with the Origin attribute being Incomplete. The following uses [Figure 1-415](#) as an example. In [Figure 1-415](#), Device A and Device B are EBGP peers, and Device B, Device C, and Device D are IBGP peers.

Figure 1-415 Networking diagram with Origin configurations



The configurations on Device D are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast
```

```
network 10.1.4.0 255.255.255.252          //Advertise the route 10.1.4.0/30.  
#
```

The configurations on Device C are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
    import-route direct          //Import direct routes.  
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table  
BGP Local router ID is 10.1.1.2  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
              h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 3  
Network          NextHop        MED      LocPrf  PrefVal Path/Ogn  
i 10.1.2.0/30    10.1.2.2      0        100     0      ?  
*>i 10.1.4.0/30  10.1.3.2      0        100     0      i  
* i             10.1.2.2      0        100     0      ?
```

The preceding command output shows that two active routes 10.1.4.0/30 are available in the routing table.

```
[~DeviceB] display bgp routing-table 10.1.4.0  
  
BGP local router ID : 10.1.1.2  
Local AS number : 65001  
Paths: 2 available, 1 best, 1 select  
BGP routing table entry information of 10.1.4.0/30:  
From: 10.1.3.2 (10.1.3.2)  
Route Duration: 01h14m48s  
Relay IP Nexthop: 0.0.0.0  
Relay IP Out-Interface: GigabitEthernet1/0/0  
Original nexthop: 10.1.3.2  
Qos information : 0x0  
AS-path Nil, origin igp, MED 0, localpref 100, pref-val 0, valid, internal, best, select, active, pre 255  
Advertised to such 1 peers:  
    10.1.1.1  
BGP routing table entry information of 10.1.4.0/30:  
From: 10.1.2.2 (10.1.2.2)  
Route Duration: 01h13m20s  
Relay IP Nexthop: 0.0.0.0  
Relay IP Out-Interface: GigabitEthernet2/0/0  
Original nexthop: 10.1.2.2  
Qos information : 0x0  
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for Origin  
Not advertised to any peer yet
```

The preceding command output shows that the route learned from Device D is selected because it is imported using the **network** command and its Origin priority is higher. **Table 1-146** describes the attribute comparison of the routes learned from Device C and Device D.

Table 1-146 Attribute comparison of the routes learned from Device C and Device D

Route Attribute	Route Learned from Device C	Route Learned from Device D	Comparison
PrefVal	0	0	The same.
Local_Pref	100	100	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	-	-	The same length.
Origin	Incomplete	IGP	The route learned from Device D is optimal.

The Origin attribute can be modified using a route-policy. In the following example, a route-policy is configured on Device B to modify the Origin attribute, and the detailed configurations are as follows:

```
#  
bgp 65001  
#  
ipv4-family unicast  
peer 10.1.3.2 route-policy for_d import      //Apply import policy named for_d to the routes learned  
from 10.1.3.2 and use for_d to modify the Origin value.  
#  
route-policy for_d permit node 10          //Define the route-policy named for_d.  
apply origin incomplete           //Set the Origin type to Incomplete.
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table  
BGP Local router ID is 10.1.1.2  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 3  
Network      NextHop      MED      LocPrf      PrefVal Path/Ogn  
i 10.1.2.0/30  10.1.2.2    0        100       0       ?  
*>i 10.1.4.0/30 10.1.2.2    0        100       0       ?  
* i          10.1.3.2    0        100       0       ?
```

The preceding command output shows that the route learned from Device C becomes the optimal route.

```
[~DeviceB] display bgp routing-table 10.1.4.0  
BGP local router ID : 10.1.1.2  
Local AS number : 65001  
Paths: 2 available, 1 best, 1 select  
BGP routing table entry information of 10.1.4.0/30:
```

```

From: 10.1.2.2 (10.1.2.2)
Route Duration: 01h28m19s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, active, pre 255
Advertised to such 1 peers:
    10.1.1.1
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.3.2 (10.1.3.2)
Route Duration: 00h03m18s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface: GigabitEthernet2/0/0
Original nexthop: 10.1.3.2
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for
router ID
Not advertised to any peer yet

```

The preceding command output shows that the route learned from Device C becomes the optimal route because it has a smaller router ID. **Table 1-147** describes the attribute comparison of the routes learned from Device C and Device D.

Table 1-147 Attribute comparison of the routes learned from Device C and Device D

Route Attribute	Route Learned from Device C	Route Learned from Device D	Comparison
PrefVal	0	0	The same.
Local_Pref	100	100	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	-	-	The same.
Origin	Incomplete	Incomplete	The same.
MED	0	0	The same.
Peer type	IBGP	IBGP	The same.
IGP cost	-	-	The same.
Cluster_List	-	-	The same.
Router ID	10.1.2.2	10.1.3.2	The route learned from Device C is optimal.

?.8. MED

MED attributes of routes can be configured as required to control traffic forwarding path for the purpose of load balancing.

The MED attribute is transmitted only within an AS or between two neighboring ASs. The AS that receives the MED attribute does not advertise it to a third AS.

Similar to the cost used by an IGP, the MED is used to determine the optimal route when traffic enters an AS. When a BGP peer learns multiple routes that have the same destination address but different next hop addresses from EBGP peers, the route with the smallest MED value is selected as the optimal route if all the other attributes are the same.

Table 1-148 lists three methods used to modify the MED value.

Table 1-148 Methods to modify the MED value

Method	Usage Scenario
Run the default med command.	This method sets a default MED for all the routes that the local device advertises to its BGP peers. The default med command takes effect only with the routes imported locally using the import-route command and BGP summarized routes.
Configure an import or export policy and run the apply cost command to configure an apply clause for the policy.	This method sets different MED values for different routes advertised by the local device to its EBGP peers.
Configure an export policy in which the apply cost-type internal command is run.	The router sets the MED of each BGP route advertised to a peer to the cost of the IGP route to which the BGP route recurses. This method takes effect only on EBGP peers. NOTE If both the apply cost and apply cost-type commands are run, only the apply cost command takes effect.
Configure an export policy in which the apply cost-type internal-inc-ibgp command is run.	The router sets the MED of each BGP route advertised to a peer to the cost of the IGP route to which the BGP route recurses. This method takes effect both on IBGP and EBGP peers.
Configure an export policy in which the apply cost-type med-plus-igp command is run.	The router sets the MED of each BGP route advertised to a peer to the cost of the IGP route to which the BGP route recurses plus the original MED. This method takes effect both on IBGP and EBGP peers.

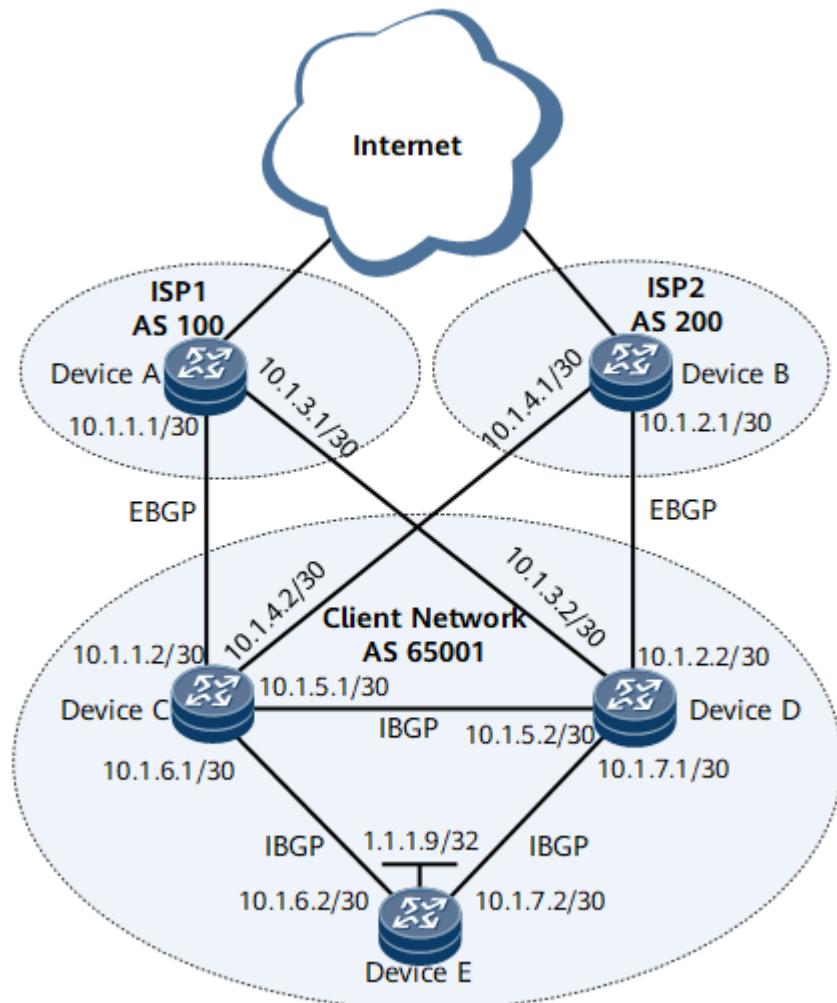
Method	Usage Scenario
Configure an export policy in which the apply cost-type med-inherit-aigp command is run.	A router sets the MED value of a route advertised to a peer to the AIGP value of the route. This applies to both IBGP and EBGP peers. This policy takes effect only for VPN BGP.

In addition, pay attention to the following rules when using the MED attribute:

- If routes are received from different ASs, traffic will be sent to different ASs. In addition, BGP selects the optimal route only from the routes destined for the same address. Therefore, BGP only compares the MEDs of routes that are from the same AS (excluding confederation sub-ASs). MEDs of two routes are compared only if the leftmost AS number in the AS_Sequence (excluding AS_Confederation_Sequence) of one route is the same as its counterpart in the other route.
- If the **compare-different-as-med** command is run, BGP compares MEDs of routes even when the routes are received from peers in different ASs. Do not run this command unless the ASs use the same IGP and route selection mode. Otherwise, a routing loop may occur.
- If a route does not carry MED, BGP uses the default value (0) as the MED of the route during route selection. If the **bestroute med-none-as-maximum** command is run, BGP considers the largest MED value (4294967295) to be the MED of the route. After route selection is complete, the MED is restored to the original value.
- After the **bestroute med-confederation** command is configured, BGP compares the MEDs of routes only when the AS_Path attributes of the routes carry no external AS numbers (ASs that do not belong to the confederation) and the leftmost AS numbers in each AS_Confederation_Sequence are the same.
- If the **deterministic-med** command is run, routes are no longer selected in the sequence in which they are received.

[Figure 1-416](#) is used as an example to describe how the MED attribute is used in BGP route selection. In [Figure 1-416](#), ISP1 and ISP2 can access 1.1.1.9/32 from DeviceC or DeviceD.

Figure 1-416 Networking diagram for MED applications



Scenario 1: Check the BGP routing tables of Device A and Device B before Device C and Device D are configured to modify the MED of the route 1.1.1.9/32.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
  Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
*>  1.1.1.9/32    10.1.1.2            0       65001i
*   10.1.3.2          0       65001i

[~DeviceB] display bgp routing-table
BGP Local router ID is 10.1.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
  Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
*>  1.1.1.9/32    10.1.4.2            0       65001i
*   10.1.2.2          0       65001i
```

The preceding command output shows that both ISP1 and ISP2 select the route learned from DeviceC as the optimal route. That is, the traffic from ISP1 and ISP2 to 1.1.1.9/32 enters AS 65001 through DeviceC, not through DeviceD, indicating that load balancing is not implemented.

Run the **display bgp routing-table ip-address** command on Device B to check the reason why Device B chooses the route learned from Device C.

```
[~DeviceB] display bgp routing-table 1.1.1.9
BGP local router ID : 10.1.2.1
Local AS number : 200
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.4.2 (10.1.1.2)
Route Duration: 00h00m58s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.4.2
Qos information : 0x0
AS-path 65001, origin igp, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.2.2
    10.1.4.2
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 00h01m07s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 65001, origin igp, pref-val 0, valid, external, pre 255, not preferred for router ID
Not advertised to any peer yet
```

The preceding command output shows that DeviceB selects the route learned from DeviceC because the router ID (10.1.1.2) of DeviceC is smaller than that (10.1.2.2) of DeviceD. **Table 1-149** describes the attribute comparison of the routes learned from DeviceC and DeviceD.

Table 1-149 Attribute comparison of the routes that DeviceB learns from DeviceC and DeviceD

Route Attribute	Route Learned from Device C	Route Learned from Device D	Comparison
PrefVal	0	0	The same.
Local_Pref	-	-	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	65001	65001	The same length.
Origin	IGP	IGP	The same.
MED	-	-	The same.
Peer type	EBGP	EBGP	The same.
IGP cost	-	-	The same.
Cluster_List	-	-	The same length.

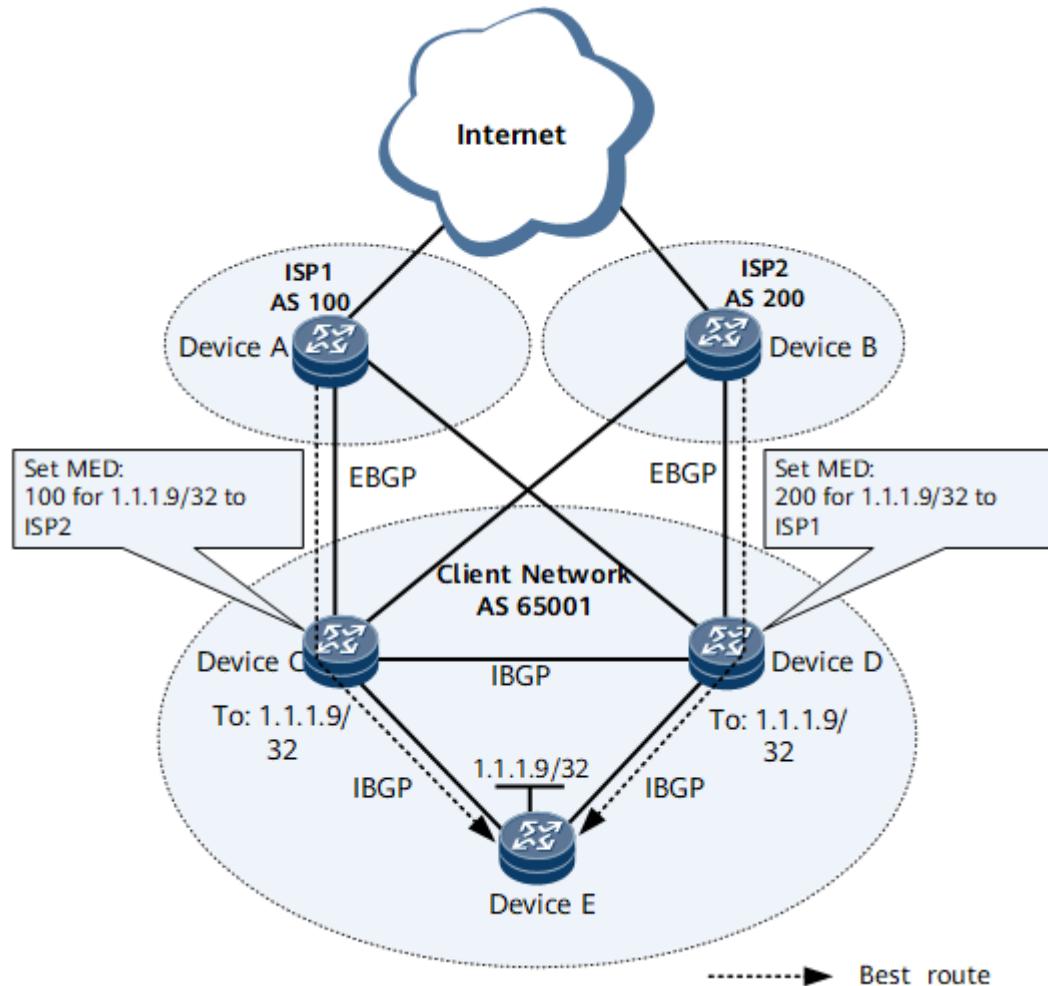
Route Attribute	Route Learned from Device C	Route Learned from Device D	Comparison
Router ID	10.1.1.2	10.1.2.2	The route learned from Device C is optimal.

Scenario 2: The requirements of the administrator of AS 65001 are as follows:

- The traffic from ISP1 to 1.1.1.9/32 enters AS 65001 preferentially through DeviceC, with DeviceD as a backup option.
- The traffic from ISP2 to 1.1.1.9/32 enters AS 65001 preferentially through DeviceD, with DeviceC as a backup option.

To meet the preceding requirements, ensure that ISP1 selects the route learned from DeviceC to access 1.1.1.9/32 and that ISP2 selects the route learned from DeviceD to access 1.1.1.9/32. **Figure 1-417** shows the networking.

Figure 1-417 Networking diagram for MED applications



The MED attribute can be used to meet the preceding requirement. However, the MED attribute needs to be set for different peers. Therefore, a route-policy must be used. Detailed configurations are as follows:

Configurations on Device C:

```
#  
bgp 65001  
#  
ipv4-family unicast  
undo synchronization  
peer 10.1.4.1 route-policy addmed100 export      //Apply export policy named addmed100 to the routes  
to be advertised to 10.1.4.1 and use addmed100 to modify the MED value.  
#  
route-policy addmed100 permit node 10           //Define the first node of addmed100 and set the MED  
of the route 1.1.1.9/32 to 100.  
if-match ip-prefix p1  
apply cost 100  
#  
route-policy addmed100 permit node 20           //Define the second node of addmed100 to allow  
addmed100 to permit all other routes.  
#  
ip ip-prefix p1 index 10 permit 1.1.1.9 32      //Configure an IP prefix list to match the route 1.1.1.9/32.
```

Configurations on Device D:

NOTE

This step is performed to ensure that Device A selects the route advertised by Device C. The preceding information, however, shows that Device A has selected the route advertised by Device C because the router ID of Device C is smaller than that of Device D. Therefore, the configuration of Device D is optional.

```
#  
bgp 65001  
#  
ipv4-family unicast  
undo synchronization  
peer 10.1.3.1 route-policy addmed200 export      //Apply the export policy named addmed200 to the  
routes to be advertised to 10.1.3.1 and use addmed200 to modify the MED value.  
#  
route-policy addmed200 permit node 10           //Define the first node of addmed200 and set the  
MED of the route 1.1.1.9/32 to 200.  
if-match ip-prefix p1  
apply cost 200  
#  
route-policy addmed200 permit node 20           //Define the second node for the route-policy  
addmed200, with no matching conditions specified, to permit all other routes.  
#  
ip ip-prefix p1 index 10 permit 1.1.1.9 32      //Configure an IP prefix list to match the route 1.1.1.9/32.
```

Run the **display bgp routing-table [ip-address]** command to check the configurations. Use Device B as an example.

Display the routing table of Device B.

```
[~DeviceB] display bgp routing-table  
BGP Local router ID is 10.1.2.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
              h - history, i - internal, s - suppressed, S - Stale  
              Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 2  
          Network      NextHop      MED      LocPrf  PrefVal Path/Ogn  
*>  1.1.1.9/32    10.1.2.2        0      65001i  
*            10.1.4.2     100        0      65001i
```

Display detailed information about the route 1.1.1.9/32 on Device B.

```
[~DeviceB] display bgp routing-table 1.1.1.9 32
BGP local router ID : 10.1.2.1
Local AS number : 200
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 01h20m38s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 65001, origin igp, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.2.2
    10.1.4.2
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.4.2 (10.1.1.2)
Route Duration: 01h16m28s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.4.2
Qos information : 0x0
AS-path 65001, origin igp, MED 100, pref-val 0, valid, external, pre 255, not preferred for MED
Not advertised to any peer yet
```

The preceding command output shows that two routes 1.1.1.9/32 are available in the BGP routing table of DeviceB and that only the route with next hop address 10.1.2.2 is selected as the optimal route. The other route with next hop address 10.1.4.2 is not selected due to the MED issue.

Table 1-150 compares the attributes of the routes that DeviceB learns from DeviceC and DeviceD.

Table 1-150 Attribute comparison of the routes that DeviceB learns from DeviceC and DeviceD

Route Attribute	Route Learned from Device C	Route Learned from Device D	Comparison
PrefVal	0	0	The same.
Local_Pref	-	-	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	65001	65001	The same length.
Origin	IGP	IGP	The same.

Route Attribute	Route Learned from Device C	Route Learned from Device D	Comparison
MED	100	-	The route learned from Device D carries no MED value, and therefore, the default value (0) is used. The route learned from DeviceD is optimal.

Figure 1-417 shows that the route learned from DeviceD does not carry the MED attribute. In this case, the default MED value 0 is used by this route. Therefore, this route is selected as the optimal route. To change the route selection result on DeviceB, you can run the **bestroute med-none-as-maximum** command on DeviceB. Detailed configurations are as follows:

```
[~DeviceB] display bgp routing-table
BGP Local router ID is 10.1.2.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
               h - history, i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
      Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
* >  1.1.1.9/32    10.1.4.2    100          0      65001i
*           10.1.2.2        0      65001i

[~DeviceB] display bgp routing-table 1.1.1.9
BGP local router ID : 10.1.2.1
Local AS number : 200
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.4.2 (10.1.1.2)
Route Duration: 00h08m42s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.4.2
Qos information : 0x0
AS-path 65001, origin igp, MED 100, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
  10.1.2.2
  10.1.4.2
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 16h33m10s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 65001, origin igp, pref-val 0, valid, external, pre 255, not preferred for MED
Not advertised to any peer yet
```

The preceding command output shows that two routes 1.1.1.9/32 are available in the BGP routing table of Device B. The MED of the route with the next hop address 10.1.4.2 is 100, and the MED of the route with the next hop address 10.1.2.2 is considered as 4294967295 because it carries no MED. Therefore, the route with the next hop address 10.1.4.2 is selected as the optimal route.

In addition, BGP selects routes in the same sequence they are received. Therefore, the route selection result is relevant to the sequence in which the routes are received. For example, the following three BGP routes are available on a device:

- Route A1: AS_Path { 65001 200 }, med 100, igr cost 13, internal, Router id 4.4.4.4
- Route A2: AS_Path { 65001 100 }, med 150, igr cost 11, internal, Router id 5.5.5.5
- Route B: AS_Path { 65002 300 }, med 0, igr cost 12, internal, Router id 6.6.6.6

If the **compare-different-as-med (BGP)** command is run, route B is the optimal route, regardless of the sequence in which the routes are received. If the **compare-different-as-med (BGP)** command is not configured, BGP does not compare the MED values of routes learned from different ASs. The route selection is described in the following cases:

- Case 1: Route A1 is received first, followed by route B, and then route A2.
 - BGP first compares route A1 and route B. Because the leftmost ASs of route A1 and route B are different, the device does not compare the MEDs of the two routes and prefers the route (route B) with the smallest IGP metric (IGP cost).
 - BGP then compares route A2 and route B. Because the leftmost ASs of route A2 and route B are different, the device does not compare the MEDs of the two routes and prefers the route (route A2) with the smallest IGP metric.
- Case 2: Route A2 is received first, followed by route B, and then route A1.
 - BGP then compares route A2 and route B. Because the leftmost ASs of route B and route A2 are different, the device does not compare the MEDs of the two routes and prefers the route (route A2) with the smallest IGP metric.
 - BGP then compares route A1 and route A2. The leftmost AS number of route A1 is the same as its counterpart in route A2. In this situation, BGP selects route A1 as the optimal route because its MED value is smaller.
- Case 3: If the **deterministic-med** command is run, BGP groups the routes that are learned from different ASs but are destined for the same network segment based on the leftmost AS number in the AS_Path, selects one optimal route from each group, and then compares the optimal routes of all the groups. Detailed steps are as follows:
 - BGP first compares route A1 and route A2. The leftmost AS number of route A1 is the same as its counterpart in route A2. In this situation, BGP selects route A1 as the optimal route because its MED value is smaller.
 - BGP then compares route A1 and route B. The leftmost AS number of route A1 is different from its counterpart in route B. Therefore, BGP does not compare the MED values and selects route B as the optimal route because the IGP cost is smaller.

Case 1 and case 2 show that the route selection result is relevant to the sequence in which routes are received if the **deterministic-med** is not configured. Case 3 shows that the route selection result is irrelevant to the sequence in which routes are received if the **deterministic-med** is configured.

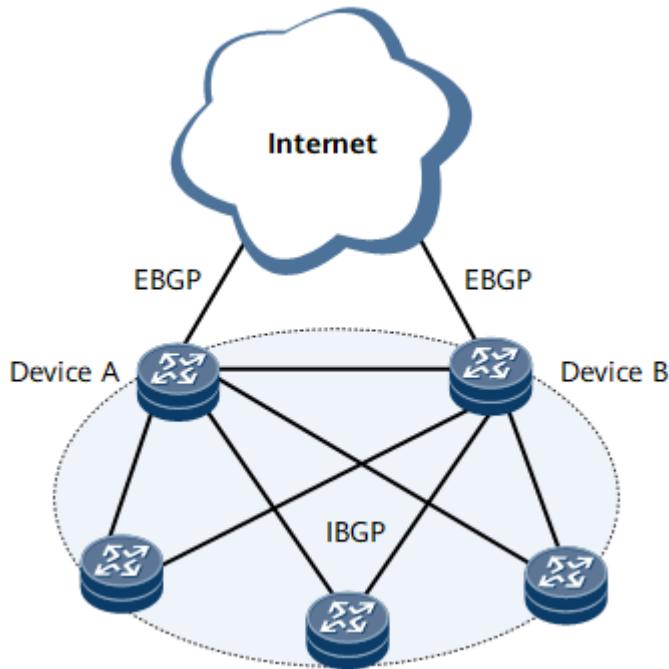
?.9. Peer Type

When IBGP routes (routes learned from IBGP peers) and EBGP routes (routes learned from EBGP peers) are available, BGP preferentially selects EBGP routes.

When one EBGP route and multiple IBGP routes are available, BGP selects the optimal route based on the peer type. If no EBGP route is available or multiple EBGP routes are available, BGP is unable to select the optimal route based on the peer type.

When multiple egress devices reside on a carrier network and receive routes from the Internet, the egress devices select the optimal route based on the peer type in most cases. In [Figure 1-418](#), all devices reside in the same AS, Device A and Device B function as egress devices and are IBGP peers of each other and of other devices in the AS. In addition, Device A and Device B receive routes from the Internet and advertise EBGP routes to all their IBGP peers. In this case, Device A and Device B each have an IBGP route and an EBGP route, and the AS_Path attributes of the two routes are the same. In this situation, Device A and Device B select the EBGP route as the optimal route.

Figure 1-418 Peer type application networking



The EBGP route is selected as the optimal route, which prevents the traffic that leaves Device A or Device B for the Internet from being forwarded to the other egress device.

For more peer type-based route selection examples, see [Local_Pref](#).

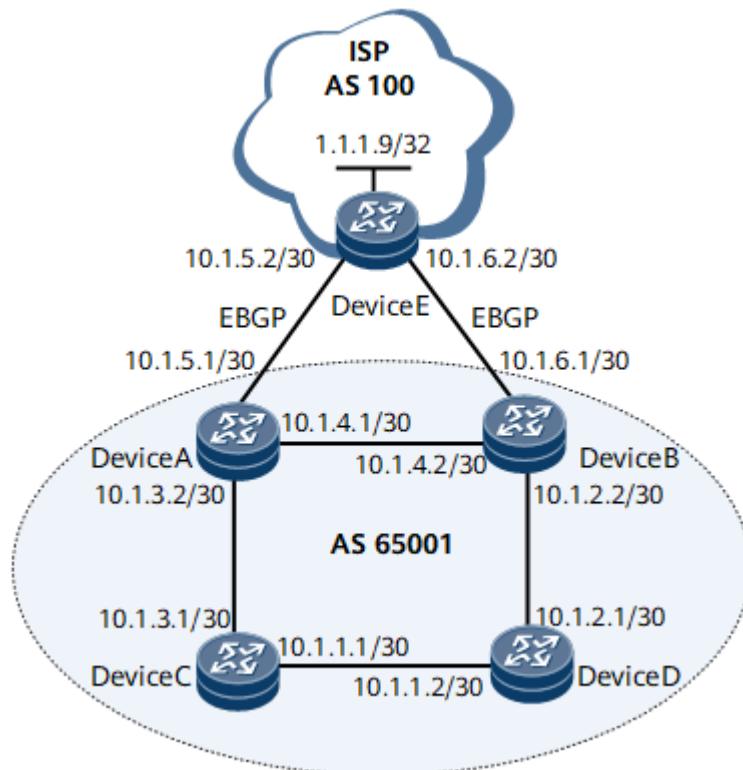
?.10. IGP Cost

BGP prefers the route with the smallest IGP cost during BGP route selection.

This rule helps BGP to choose the route with the smallest cost to its recursive next hop address quickly. If the **bestroute igrp-metric-ignore** command is configured,

BGP does not compare the IGP cost. In [Figure 1-419](#), OSPF runs in AS 65001, an EBGP peer relationship is established between Device E and Device A and between Device E and Device B, and an IBGP peer relationship is established between Device A and Device C, between Device A and Device D, between Device B and Device C, and between Device B and Device D; Device E is configured to import routes (1.1.1.9/32 for example) from AS 100 to BGP.

Figure 1-419 Networking diagram with IGP cost configurations



Run the **display bgp routing-table [ip-address]** command on Device C and Device D to check the configurations. Device C is used as an example.

Display the routing table of Device C.

```
[~DeviceC] display bgp routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 4
      Network          NextHop        MED     LocPrf  PrefVal Path/Ogn
    *>i 1.1.1.9/32    10.1.5.2      0       100      0   100i
    * i 10.1.6.2/30   10.1.6.2      0       100      0   100i
    *>i 10.1.5.0/30   10.1.3.2      0       100      0   i
    *>i 10.1.6.0/30   10.1.2.2      0       100      0   i
```

The preceding command output shows that two routes 1.1.1.9/32 are available in the routing table of Device C and that Device C selects the route learned from Device A.

```
[~DeviceC] display bgp routing-table 1.1.1.9
```

```
BGP local router ID : 10.1.1.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.3.2 (2.2.2.9)
Route Duration: 00h00m44s
Relay IP Nexthop: 10.1.3.2
Relay IP Out-Interface: GigabitEthernet2/0/0
Original nexthop: 10.1.5.2
Qos information : 0x0
AS-path 100, origin igp, MED 0, localpref 100, pref-val 0, valid, internal, best, select, active, pre 255
Not advertised to any peer yet

BGP routing table entry information of 1.1.1.9/32:
From: 10.1.2.2 (10.1.2.2)
Route Duration: 00h00m39s
Relay IP Nexthop: 10.1.1.2
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.6.2
Qos information : 0x0
AS-path 100, origin igp, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, IGP cost 2, not preferred
for IGP cost
Not advertised to any peer yet
```

The preceding command output shows that the route with next hop address 10.1.6.2 is ignored because its IGP cost is larger than that of the other route. **Table 1-151** describes the attribute comparison of the routes learned from Device A and Device B.

Table 1-151 Attribute comparison of the routes learned from Device A and Device B.

Route Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
PrefVal	0	0	The same.
Local_Pref	100	100	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	100	100	The same length.
Origin	IGP	IGP	The same.
MED	0	0	The same.
Peer type	IBGP	IBGP	The same.

Route Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
IGP cost	-	2	<p>The route learned from Device A is optimal.</p> <p>NOTE If a BGP route carries no IGP cost value, BGP considers its IGP cost to be 0. If no IGP routes are used during BGP peer relationship establishment or the costs of used IGP routes are 0, the IGP cost is not displayed in the display bgp routing-table ip-address command output.</p>

?11. Cluster_List

BGP prefers the route with the shortest Cluster_List length during BGP route selection.

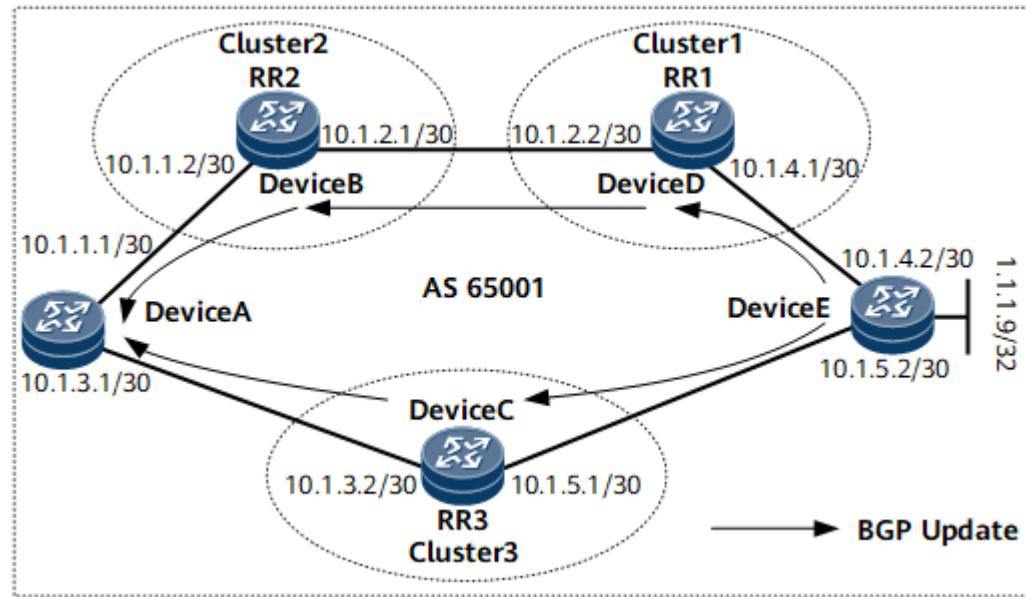
An RR and its clients form a cluster. In an AS, each RR is uniquely identified by a Cluster_ID.

Similar to an AS_Path, a Cluster_List is composed of a series of Cluster_IDs and is generated by an RR. The Cluster_List records all the RRs through which a route passes.

- Before an RR reflects a route between its clients or between its clients and non-clients, the RR adds the local Cluster_ID to the leftmost position of the Cluster_List. If a route does not carry any Cluster_List, the RR creates one for the route.
- After the RR receives an updated route, it checks the Cluster_List of the route. If the RR finds that its cluster ID is included in the Cluster_List, the RR discards the route. If its cluster ID is not included in the Cluster_List, the RR adds its cluster ID to the Cluster_List and then reflects the route.
- The RR does not add the Cluster_ID attribute to the routes that are locally imported.

The following example shows how Cluster_List is used in BGP route selection. In **Figure 1-420**, an IBGP peer relationship is established between each two neighboring devices in AS 65001. DeviceB functions as a level-1 RR, and DeviceD is its client. DeviceD functions as a level-2 RR, and DeviceE is its client. DeviceC functions as an RR, and DeviceE is its client. DeviceE is configured to import the route 1.1.1.9/32 to BGP.

Figure 1-420 Networking diagram with Cluster_List configurations



Run the **display bgp routing-table [ip-address]** command on Device A to check the configurations.

Display the routing table of Device A.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.3.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
      Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
*>i 1.1.1.9/32      10.1.5.2       0        100      0      i
* i           10.1.4.2       0        100      0      i
```

The preceding command output shows that two routes 1.1.1.9/32 are available in the routing table and that Device A selects the route learned from Device C.

```
[~DeviceA] display bgp routing-table 1.1.1.9
BGP local router ID : 10.1.3.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 1.1.1.9/32:
From: 10.1.3.2 (2.2.2.9)
Route Duration: 00h53m08s
Relay IP Nexthop: 10.1.3.2
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.5.2
Qos information : 0x0
AS-path Nil, origin igp, MED 0, localpref 100, pref-val 0, valid, internal, best, select, active, pre 255, IGP cost 3
Originator: 1.1.1.9
Cluster list: 0.0.0.3
Not advertised to any peer yet

BGP routing table entry information of 1.1.1.9/32:
From: 10.1.1.2 (10.1.2.1)
Route Duration: 00h28m05s
Relay IP Nexthop: 10.1.1.2
```

```

Relay IP Out-Interface: GigabitEthernet2/0/0
Original nexthop: 10.1.4.2
Qos information : 0x0
AS-path Nil, origin igp, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, IGP cost 3, not preferred for Cluster List
Originator: 1.1.1.9
Cluster list: 0.0.0.2, 0.0.0.1
Not advertised to any peer yet

```

The preceding command output shows that the route learned from DeviceB is ignored because its Cluster_List is longer. **Table 1-152** describes attribute comparison of the routes learned by DeviceA from DeviceB and DeviceC.

Table 1-152 Attribute comparison of the routes learned by Device A from Device B and Device C

Route Attribute	Route Learned from Device B	Route Learned from Device C	Comparison
PrefVal	0	0	The same.
Local_Pref	100	100	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	-	-	The same length.
Origin	IGP	IGP	The same.
MED	0	0	The same.
Peer type	IBGP	IBGP	The same.
IGP cost	3	3	The same.
Cluster_List	0.0.0.2, 0.0.0.1	0.0.0.3	The route learned from Device C is optimal.

In most cases, BGP does not advertise the routes learned from an AS to the AS again. When RRs are deployed, such routes may be advertised to the AS again although routing loops may occur. Using the Cluster_List attribute can prevent such routing loops.

?12. Originator_ID

If routes carry the Originator_ID, the originator ID is substituted for the router ID during route selection. The route with the smallest Originator_ID is preferred.

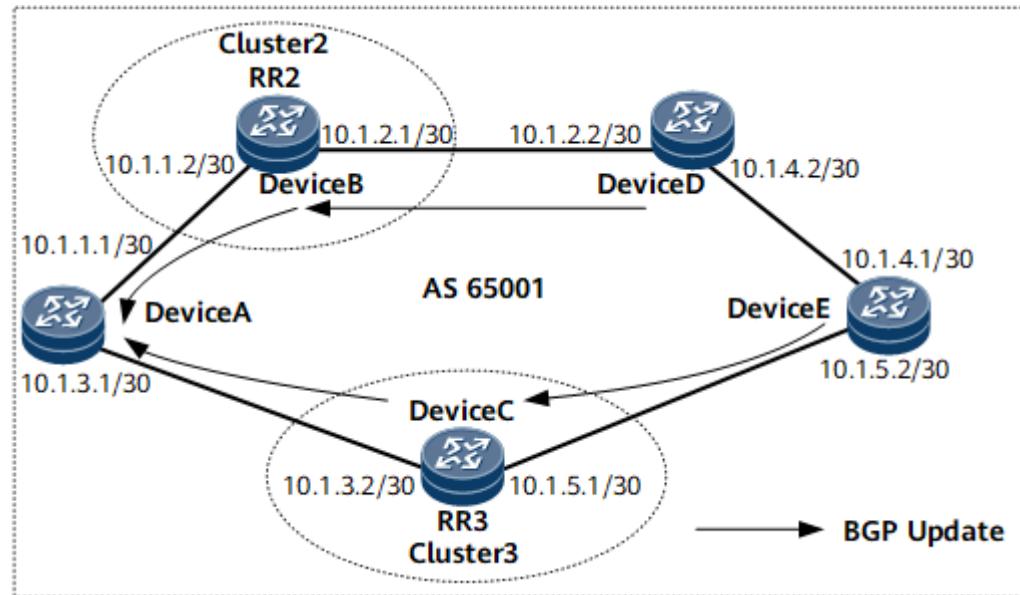
The Originator_ID attribute is four bytes long and is generated by an RR. It carries the router ID of the route originator in the local AS.

- When a route is reflected by an RR for the first time, the RR adds the Originator_ID to this route. If a route already carries the Originator_ID attribute, the RR does not create a new one.

- After receiving the route, a BGP speaker checks whether the Originator_ID is the same as its router ID. If Originator_ID is the same as its router ID, the BGP speaker discards this route.

The following example shows how Originator_ID is used during BGP route selection. In **Figure 1-421**, an IBGP peer relationship is established between each two neighboring devices in AS 65001. The router IDs of DeviceB and DeviceC are 2.2.2.9 and 3.3.3.9, respectively, and they function as RRs. DeviceD is a client of DeviceB, and DeviceE is a client of DeviceC. DeviceD and DeviceE are configured to import the route 10.1.4.0/30 to BGP.

Figure 1-421 Networking diagram with Originator_ID configurations



Run the **display bgp routing-table [ip-address]** command on Device A to check the configurations.

Display the routing table of Device A.

```
[~DeviceA] display bgp routing-table
BGP Local router ID is 10.1.3.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
              RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
      Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
    *>i 10.1.4.0/30    10.1.5.2        0     100      0      i
    * i                  10.1.2.2        0     100      0      i
```

The preceding command output shows that two routes 10.1.4.0/30 are available in the routing table of DeviceA and that DeviceA selects the route learned from DeviceC.

```
[~DeviceA] display bgp routing-table 10.1.4.0
BGP local router ID : 10.1.3.1
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.1.4.0/30:
From: 10.1.3.2 (3.3.3.9)
```

```

Route Duration: 00h00m01s
Relay IP Nexthop: 10.1.3.2
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.5.2
Qos information : 0x0
AS-path Nil, origin igrp, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255, IGP cost 2
Originator: 10.1.4.1
Cluster list: 0.0.0.3
Not advertised to any peer yet

BGP routing table entry information of 10.1.4.0/30:
From: 10.1.1.2 (2.2.2.9)
Route Duration: 00h00m17s
Relay IP Nexthop: 10.1.1.2
Relay IP Out-Interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path Nil, origin igrp, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, IGP cost 2, not preferred for router ID
Originator: 10.1.4.2
Cluster list: 0.0.0.2
Not advertised to any peer yet

```

The preceding command output shows that the route learned from DeviceB is not selected due to a router ID issue. In fact, the router ID of DeviceB is 2.2.2.9, smaller than that (3.3.3.9) of DeviceC. The route learned from DeviceB should be selected if the router IDs are used to determine the optimal route. However, the routes carry the Originator_ID attribute. In this situation, Originator_IDs (rather than router IDs) are compared. Consequently, the route learned from DeviceC is selected because the Originator_ID (10.1.4.1) of DeviceC is smaller than that (10.1.4.2) of the route learned from DeviceB.

Table 1-153 describes the attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC.

Table 1-153 Attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC

Route Attribute	Route Learned from Device B	Route Learned from Device C	Comparison
PrefVal	0	0	The same.
Local_Pref	100	100	The same.
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	-	-	The same.
AS_Path	-	-	The same length.
Origin	IGP	IGP	The same.
MED	0	0	The same.
Peer type	IBGP	IBGP	The same.
IGP cost	2	2	The same.
Cluster_List	0.0.0.2	0.0.0.3	The same length.

Route Attribute	Route Learned from Device B	Route Learned from Device C	Comparison
Originator_ID	10.1.4.2	10.1.4.1	The route learned from Device C is optimal.

If routes carry Originator_ID attributes, the Originator_ID attributes (rather than router IDs) are compared.

?.13. Router ID

If multiple routes to the same destination are available, BGP preferentially selects the route advertised by the device with the smallest router ID.

A router ID identifies a router in an AS. To configure a router ID, you can perform either of the following operations:

- Run the **router-id { ipv4-address | vpn-instance auto-select }** command in the BGP view. If the command is not run, BGP automatically selects the router ID in the system view as the router ID.
- You can run the **router-id { ipv4-address | auto-select }** command in the BGP VPN instance IPv4 or IPv6 address family view to configure a router ID. The command run in the BGP VPN instance IPv4 or IPv6 address family view takes precedence over that run in the BGP view.

In addition, pay attention to the impact of the following rule on BGP route selection: If routes carry the Originator_ID attribute, the device compares Originator_IDs rather than router IDs during route selection and prefers the route with the smallest Originator_ID. By default, BGP compares Cluster_Lists prior to Originator_IDs during route selection. To enable BGP to compare Originator_IDs prior to Cluster_Lists during route selection, run the **bestroute routerid-prior-clusterlist** command.

For more router ID-based route selection examples, see [Local_Pref](#), [Origin](#), and [MED](#).

?.14. Peer IP Address

BGP prefers the route learned from the peer with the smallest IP address during BGP route selection.

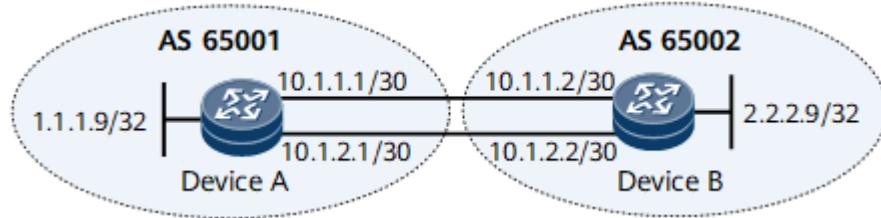
The peer IP address is the IP address specified in *ipv4-address* or *ipv6-address* in the **peer { group-name | ipv4-address | ipv6-address } as-number { as-number-plain | as-number-dot }** command. The *group-name* parameter specified in the command is the one specified in the **peer { ipv4-address | ipv6-address } group group-name** command.

If the optimal route has not been selected yet before BGP begins to compare peer IP addresses, the local device may have established multiple BGP peer relationships with another device through equal-cost links. In most cases, if a backup physical link is available between two devices, using loopback interfaces to establish a BGP peer relationship is recommended although multiple BGP peer

relationships may be established between the two devices through the physical links.

In **Figure 1-422**, two physical links are available between Device A and Device B. Device A and Device B can use loopback interfaces to establish a BGP peer relationship or use the two links to establish two BGP peer relationships. In the following example, the two links are used to establish two BGP peer relationships to show how peer addresses are used in route selection.

Figure 1-422 Networking in which two links are used to establish two BGP peer relationships



The configurations on Device A are as follows:

```
#  
bgp 65001  
peer 10.1.1.2 as-number 65002  
peer 10.1.2.2 as-number 65002  
#  
ipv4-family unicast  
peer 10.1.1.2 enable  
peer 10.1.2.2 enable  
#
```

The configurations on Device B are as follows:

```
#  
bgp 65002  
peer 10.1.1.1 as-number 65001  
peer 10.1.2.1 as-number 65001  
#  
ipv4-family unicast  
network 2.2.2.9 255.255.255.255  
peer 10.1.1.1 enable  
peer 10.1.2.1 enable  
#
```

Run the **display bgp routing-table [ip-address]** command to check the configurations.

Display the routing table of Device A.

```
[~DeviceA] display bgp routing-table  
BGP Local router ID is 192.168.2.3  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 2  
Network NextHop MED LocPrf PrefVal Path/Ogn  
*> 2.2.2.9/32 10.1.1.2 0 0 65002i  
* 10.1.2.2 0 0 65002i
```

The preceding command output shows that two routes 2.2.2.9/32 are available in the routing table and that the route with the next hop address 10.1.1.2 is selected as the optimal route.

```
[~DeviceA] display bgp routing-table 2.2.2.9
BGP local router ID : 192.168.2.3
Local AS number : 65001
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 2.2.2.9/32:
From: 10.1.1.2 (192.168.2.4)
Route Duration: 00h19m10s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 65002, origin igp, MED 0, pref-val 0, valid, external, best, select, active, pre 255
Advertised to such 2 peers:
    10.1.1.2
    10.1.2.2
BGP routing table entry information of 2.2.2.9/32:
From: 10.1.2.2 (192.168.2.4)
Route Duration: 00h19m05s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 65002, origin igp, MED 0, pref-val 0, valid, external, pre 255, not preferred for peer address
Not advertised to any peer yet
```

The preceding command output shows that the route with the next hop address 10.1.1.2 is selected as the optimal route because its peer IP address is smaller than that of the other route.

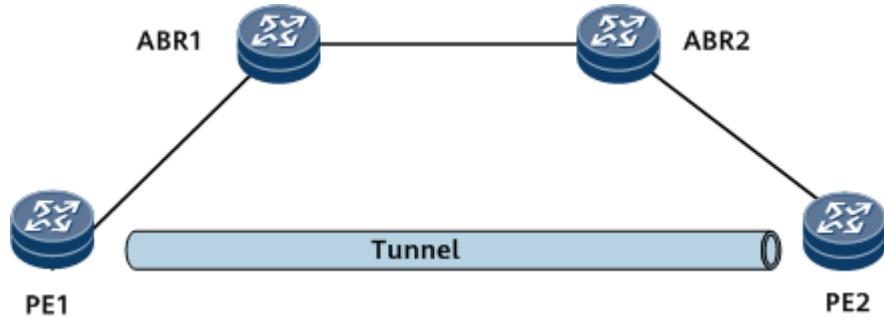
(Optional) Configuring BGP to Ignore the Reachability of the Next Hops of Received BGP VPNv4 Routes

Configuring BGP to ignore the reachability of the next hops of received BGP VPNv4 routes ensures that the BGP routes remain active even if their next hops are unreachable.

Usage Scenario

On the network shown in [Figure 1-423](#), an IGP runs between PE1 and ABR1, and between PE2 and ABR2. All the devices run IBGP. ABR1 and ABR2 are route reflectors (RRs). PE1 and ABR2 are the clients of ABR1, and PE2 and ABR1 are the clients of ABR2. A tunnel is deployed between PE1 and PE2. When ABR2 receives a BGP VPNv4 route with PE1 as the original next hop from ABR1, ABR2 cannot find the IP routing entry corresponding to PE1 and does not have a tunnel to PE1. As a result, ABR2 considers the received BGP VPNv4 route to be invalid, and this route cannot be advertised to PE2. As PE2 does not receive the BGP VPNv4 route originating from PE1, route recursion to the tunnel between PE1 and PE2 cannot be performed, and traffic cannot be forwarded through the tunnel. To address this, configure BGP on ABR2 to ignore the reachability of the next hops of received BGP VPNv4 routes. In this way, PE2 can learn the BGP VPNv4 route originating from PE1 through ABR2 properly, and the route can recurse to the tunnel for traffic forwarding.

Figure 1-423 Configuring BGP to ignore the reachability of the next hops of received BGP VPNv4 routes



Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family vpnv4**

The BGP-VPNv4 address family view is displayed.

Step 4 Run **bestroute nexthop-resolved none**

BGP on the local device is configured to ignore the reachability of the next hops of received BGP VPNv4 routes.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the **display bgp vpnv4 all routing-table network** command to check the next-hop IP address obtained through route recursion and the outbound interface of the recursive tunnel.

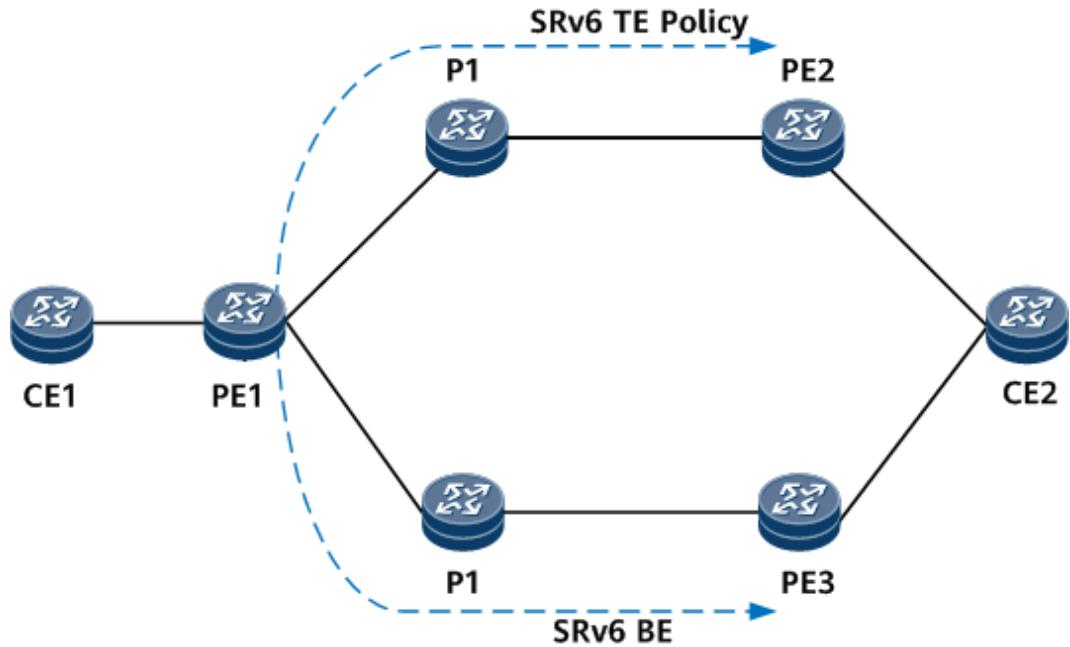
Configuring BGP to Preferentially Select the Routes with Next Hops Recursing to SRv6 TE Policies

Usage Scenario

On the network shown in [Figure 1-424](#), an SRv6 TE Policy is deployed between PE1 and PE2, and an SRv6 BE tunnel is deployed between PE1 and PE3. PE1 has two routes with the same prefix but different next hops. One of the routes recurses to the SRv6 TE Policy, and the other route recurses to the SRv6 BE tunnel. If services need to be carried over the SRv6 TE Policy, you can configure this function so that BGP compares the types of SRv6 tunnels to which route next hops recurse when selecting the optimal route. Among the routes with next hops

recursing to SRv6 Policies, those with next hops recursing to SRv6 TE Policies take precedence over those with next hops recursing to SRv6 BE tunnels.

Figure 1-424 Network diagram of configuring BGP to preferentially select the routes with next hops recursing to SRv6 TE Policies



Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp as-number**

The BGP view is displayed.

Step 3 Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

Step 4 Run **bestroute nexthop-recursive-priority srv6-te-policy**

BGP is configured to compare the types of SRv6 tunnels to which route next hops recurse when selecting the optimal route and preferentially select those with next hops recursing to SRv6 TE Policies over those with next hops recursing to SRv6 BE tunnels.

Step 5 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the **display bgp routing-table network** command to check BGP route selection results.

Configuring BGP to Preferentially Select the Prefix Routes That Are Learned from VPNv4, VPNv6, or EVPN Peers and Are Leaked to a VPN Instance

Usage Scenario

When VPNv4 and EVPN L3VPNv4 services, or VPNv6 and EVPN L3VPNv6 services coexist, a BGP IPv4/IPv6 VPN instance may have two routes with the same prefix but different VPN address family types, with one route learned from a VPNv4 or VPNv6 peer and leaked to a VPN instance, and the other learned from an EVPNv4 or EVPNv6 peer and leaked to a VPN instance. In this case, you can configure BGP to preferentially select either type of the preceding routes.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip vpn-instance *vpn-instance-name***

A VPN instance is created and its view is displayed.

Step 3 Run **ipv4-family**

The VPN instance IPv4 address family is enabled, and the view of this address family is displayed.

Or run **ipv6-family**

The VPN instance IPv6 address family is enabled, and the view of this address family is displayed.

Step 4 Run **route-distinguisher *route-distinguisher***

An RD is configured.

Step 5 Run **vpn-target *vpn-target &<1-8>* [both | export-extcommunity | import-extcommunity]**

VPN targets are configured.

Step 6 Run **quit**

Exit the VPN instance IPv4 address family view or VPN instance IPv6 address family view.

Step 7 Run **quit**

Exit the VPN instance view.

Step 8 Run **bgp *as-number***

The BGP view is displayed.

Step 9 Run **ipv4-family *vpn-instance *vpn-instance-name****

The BGP VPN instance IPv4 address family view is displayed.

Or run **ipv6-family *vpn-instance *vpn-instance-name****

The BGP VPN instance IPv6 address family view is displayed.

Step 10 Perform the following configurations as required:

- To configure BGP to compare the VPN address family types of routes when selecting the optimal route and preferentially select the IPv4 prefix routes leaked from VPNV4 over those leaked from EVPNV4, run the **bestroute address-family-priority vpnv4 high-level** command in the BGP VPN instance IPv4 address family view.
- To configure BGP to compare the VPN address family types of routes when selecting the optimal route and preferentially select the IPv4 prefix routes leaked from EVPNV4 over those leaked from VPNV4, run the **bestroute address-family-priority evpnv4 high-level** command in the BGP VPN instance IPv4 address family view.
- To configure BGP to compare the VPN address family types of routes when selecting the optimal route and preferentially select the IPv6 prefix routes leaked from VPNV6 over those leaked from EVPNV6, run the **bestroute address-family-priority vpnv6 high-level** command.
- To configure BGP to compare the VPN address family types of routes when selecting the optimal route and preferentially select the IPv6 prefix routes leaked from EVPNV6 over those leaked from VPNV6, run the **bestroute address-family-priority evpnv6 high-level** command.

 **NOTE**

If there are a large number of BGP IPv4 VPN instances and the **bestroute address-family-priority { vpnv4 | evpnv4 } high-level** command needs to be run for each instance, you can run the **bestroute address-family-priority { vpnv4 | evpnv4 } high-level all-vpn-instance** command instead to simplify the configuration because the latter command takes effect for all BGP IPv4 VPN address families.

If there are a large number of BGP IPv6 VPN instances and the **bestroute address-family-priority { vpnv6 | evpnv6 } high-level** command needs to be run for each instance, you can run the **bestroute address-family-priority { vpnv6 | evpnv6 } high-level all-vpn-instance** command instead to simplify the configuration because the latter command takes effect for all BGP IPv6 VPN address families.

Step 11 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

- Run the **display bgp vpnv4 vpn-instance vpn-instance-name routing-table ipv4-address [mask-length | mask]** command to check the BGP routes of an IPv4 VPN instance.
- Run the **display bgp vpnv6 vpn-instance vpn-instance-name routing-table ipv6-address [prefix-length]** command to check the BGP routes of an IPv6 VPN instance.

1.1.9.2.58 Configuration Examples for BGP

This section provides BGP configuration examples.

Example for Configuring Basic BGP Functions

Before building BGP networks, you need to configure basic BGP functions.

Networking Requirements

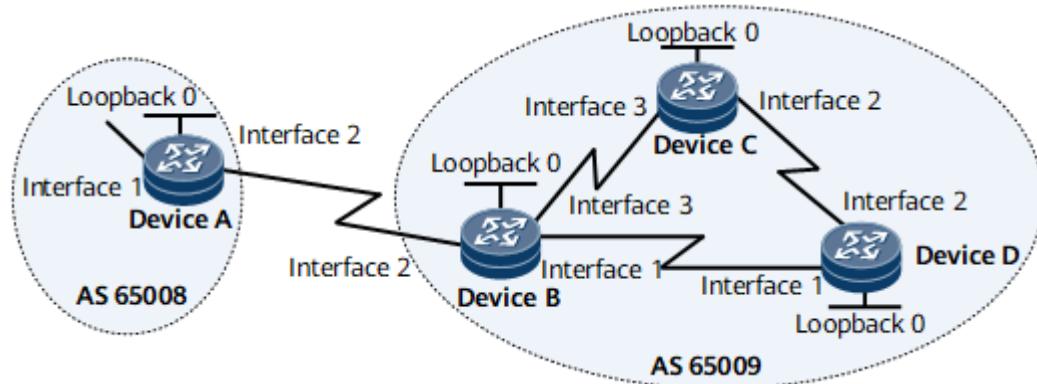
If multiple ASs want to access each other, these ASs must exchange their local routes. If multiple routers exist in the ASs, a great deal of routing information will be exchanged between ASs, which consumes lots of bandwidth resources. To address this issue, you can configure basic BGP functions.

In [Figure 1-425](#), DeviceA is in AS 65008. DeviceB, DeviceC, and DeviceD are in AS 65009. The routing tables of these routers store many routes, and the routes change frequently. After BGP is enabled on the routers, they can exchange routing information. If routes of one router changes, the router sends Update messages carrying only changed routing information to its peers, which greatly reduces bandwidth consumption.

Figure 1-425 Configuring basic BGP functions

 NOTE

Interfaces 1 through 3 in this example are GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



Device Name	Interface	IP Address
DeviceA	Loopback 0	1.1.1.1/32
	GE 1/0/0	172.16.0.1/16
	GE 2/0/0	192.168.0.1/24
DeviceB	Loopback 0	2.2.2.2/32
	GE 1/0/0	10.1.1.1/24
	GE 2/0/0	192.168.0.2/24
	GE 3/0/0	10.1.3.1/24
DeviceC	Loopback 0	3.3.3.3/32
	GE 2/0/0	10.1.2.1/24
	GE 3/0/0	10.1.3.2/24
DeviceD	Loopback 0	4.4.4.4/32
	GE 1/0/0	10.1.1.2/24

Device Name	Interface	IP Address
	GE 2/0/0	10.1.2.2/24

Precautions

When configuring basic BGP functions, note the following rules:

- If the peer IP address specified during peer relationship establishment is a loopback interface address or a sub-interface IP address, you need to run the **peer connect-interface** command on both ends to ensure that the two ends are correctly connected.
- If there is no directly connected physical link between EBGP peers, run the **peer ebgp-max-hop** command to allow EBGP peers to establish TCP connections through multiple hops.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

Configuration Roadmap

The configuration roadmap is as follows:

1. Establish IBGP connections between DeviceB, DeviceC, and DeviceD.
2. Establish an EBGP connection between DeviceA and DeviceB.
3. Advertise routes using the **network** command on DeviceA, and then check the routing tables of DeviceA, DeviceB, and DeviceC.
4. Configure BGP on DeviceB to import direct routes, and then check the routing tables of DeviceA and DeviceC.

Data Preparation

To complete the configuration, you need the following data:

- Router ID and AS number of DeviceA
- Router IDs and AS numbers of DeviceB, DeviceC, and DeviceD

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure OSPF.

Configure DeviceB.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 10.1.3.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 2.2.2.2 0.0.0.0
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
```

```
[~DeviceB-ospf-1] quit
```

```
# Configure DeviceC.
```

```
[~DeviceC] ospf 1
```

```
[*DeviceC-ospf-1] area 0
```

```
[*DeviceC-ospf-1-area-0.0.0.0] network 10.1.2.0 0.0.0.255
```

```
[*DeviceC-ospf-1-area-0.0.0.0] network 10.1.3.0 0.0.0.255
```

```
[*DeviceC-ospf-1-area-0.0.0.0] network 3.3.3.3 0.0.0.0
```

```
[*DeviceC-ospf-1-area-0.0.0.0] commit
```

```
[~DeviceC-ospf-1-area-0.0.0.0] quit
```

```
[~DeviceC-ospf-1] quit
```

```
# Configure DeviceD.
```

```
[~DeviceD] ospf 1
```

```
[*DeviceD-ospf-1] area 0
```

```
[*DeviceD-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
```

```
[*DeviceD-ospf-1-area-0.0.0.0] network 10.1.2.0 0.0.0.255
```

```
[*DeviceD-ospf-1-area-0.0.0.0] network 4.4.4.4 0.0.0.0
```

```
[*DeviceD-ospf-1-area-0.0.0.0] commit
```

```
[~DeviceD-ospf-1-area-0.0.0.0] quit
```

```
[~DeviceD-ospf-1] quit
```

Step 3 Configure IBGP connections.

```
# Configure DeviceB.
```

```
[~DeviceB] bgp 65009
```

```
[*DeviceB-bgp] router-id 2.2.2.2
```

```
[*DeviceB-bgp] peer 3.3.3.3 as-number 65009
```

```
[*DeviceB-bgp] peer 4.4.4.4 as-number 65009
```

```
[*DeviceB-bgp] peer 3.3.3.3 connect-interface LoopBack0
```

```
[*DeviceB-bgp] peer 4.4.4.4 connect-interface LoopBack0
```

```
[*DeviceB-bgp] commit
```

```
[~DeviceB-bgp] quit
```

```
# Configure DeviceC.
```

```
[~DeviceC] bgp 65009
```

```
[*DeviceC-bgp] router-id 3.3.3.3
```

```
[*DeviceC-bgp] peer 2.2.2.2 as-number 65009
```

```
[*DeviceC-bgp] peer 4.4.4.4 as-number 65009
```

```
[*DeviceC-bgp] peer 2.2.2.2 connect-interface LoopBack0
```

```
[*DeviceC-bgp] peer 4.4.4.4 connect-interface LoopBack0
```

```
[*DeviceC-bgp] commit
```

```
[~DeviceC-bgp] quit
```

```
# Configure DeviceD.
```

```
[~DeviceD] bgp 65009
```

```
[*DeviceD-bgp] router-id 4.4.4.4
```

```
[*DeviceD-bgp] peer 2.2.2.2 as-number 65009
```

```
[*DeviceD-bgp] peer 3.3.3.3 as-number 65009
```

```
[*DeviceD-bgp] peer 2.2.2.2 connect-interface LoopBack0
```

```
[*DeviceD-bgp] peer 3.3.3.3 connect-interface LoopBack0
```

```
[*DeviceD-bgp] commit
```

```
[~DeviceD-bgp] quit
```

Step 4 Configure an EBGP connection.

```
# Configure DeviceA.
```

```
[~DeviceA] bgp 65008
```

```
[*DeviceA-bgp] router-id 1.1.1.1
```

```
[*DeviceA-bgp] peer 192.168.0.2 as-number 65009
```

```
[*DeviceA-bgp] commit
```

```
[~DeviceA-bgp] quit
```

```
# Configure DeviceB.
```

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] peer 192.168.0.1 as-number 65008
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

Check the status of BGP connections.

```
[~DeviceB] display bgp peer

BGP local router ID : 2.2.2.2
Local AS number : 65009
Total number of peers : 3          Peers in established state : 3

Peer      V   AS MsgRcvd MsgSent OutQ Up/Down    State PrefRcv
3.3.3.3    4 65009     5     5  0 00:44:58 Established    0
4.4.4.4    4 65009     4     4  0 00:40:54 Established    0
192.168.0.1 4 65008     3     3  0 00:44:03 Established    0
```

The command output shows that DeviceB has established BGP connections with other routers and that the connection status is Established.

Step 5 Configure DeviceA to advertise the route to 172.16.0.0/16.

Configure DeviceA to advertise the route.

```
[~DeviceA] bgp 65008
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] network 172.16.0.0 255.255.0.0
[*DeviceA-bgp-af-ipv4] network 192.168.0.0 255.255.255.0
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

Check the routing table of DeviceA.

```
[~DeviceA] display bgp routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
*> 172.16.0.0  0.0.0.0    0        0          i
```

Check the routing table of DeviceB.

```
[~DeviceB] display bgp routing-table

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network      NextHop      MED      LocPrf      PrefVal Path/Ogn
*> 172.16.0.0  192.168.0.1  0        0        65008i
```

Check the routing table of DeviceC.

```
[~DeviceC] display bgp routing-table
```

BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1

Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
i 172.16.0.0	192.168.0.1	0	100	0	65008i

NOTE

The command output shows that DeviceC has learned the route to 172.16.0.0 from AS 65008. However, this route is invalid because the next hop 192.168.0.1 is unreachable.

Step 6 Configure BGP to import direct routes.

Configure DeviceB.

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] import-route direct
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
[~DeviceB-bgp] quit
```

Check the BGP routing table of DeviceA.

```
[~DeviceA] display bgp routing-table
```

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 5

Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
*> 2.2.2.2/32	192.168.0.2	0	0	65009?	
*> 172.16.0.0	0.0.0.0	0	0	i	
*> 10.1.1.0/24	192.168.0.2	0	0	65009?	
*> 10.1.3.0/24	192.168.0.2	0	0	65009?	
*> 192.168.0.0	192.168.0.2	0	0	65009?	

Check the routing table of DeviceC.

```
[~DeviceC] display bgp routing-table
```

BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 5

Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
i 2.2.2.2/32	2.2.2.2	0	100	0	?
*>i 10.1.1.0/24	2.2.2.2	0	100	0	?
* i 10.1.3.0/24	2.2.2.2	0	100	0	?
*>i 172.16.0.0	192.168.0.1	0	100	0	65008i
*>i 192.168.0.0	2.2.2.2	0	100	0	?

The command output shows that the route to 172.16.0.0 becomes valid and that the next hop is the address of DeviceA.

Verify the configuration using the **ping** command.

```
[~DeviceC] ping 172.16.0.1
PING 172.16.0.1: 56 data bytes, press CTRL_C to break
Reply from 172.16.0.1: bytes=56 Sequence=1 ttl=254 time=31 ms
Reply from 172.16.0.1: bytes=56 Sequence=2 ttl=254 time=47 ms
Reply from 172.16.0.1: bytes=56 Sequence=3 ttl=254 time=31 ms
Reply from 172.16.0.1: bytes=56 Sequence=4 ttl=254 time=16 ms
Reply from 172.16.0.1: bytes=56 Sequence=5 ttl=254 time=31 ms
--- 172.16.0.1 ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 16/31/47 ms
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.0.1 255.255.0.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.0.1 255.255.255.0
#
interface LoopBack0
ip address 1.1.1.1 255.255.255.255
#
bgp 65008
router-id 1.1.1.1
peer 192.168.0.2 as-number 65009
#
ipv4-family unicast
undo synchronization
network 172.16.0.0 255.255.0.0
network 192.168.0.0 255.255.255.0
peer 192.168.0.2 enable
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0
#
interface LoopBack0
ip address 2.2.2.2 255.255.255.255
#
bgp 65009
router-id 2.2.2.2
peer 3.3.3.3 as-number 65009
```

```
peer 3.3.3.3 connect-interface LoopBack0
peer 4.4.4.4 as-number 65009
peer 4.4.4.4 connect-interface LoopBack0
peer 192.168.0.1 as-number 65008
#
ipv4-family unicast
undo synchronization
import-route direct
peer 3.3.3.3 enable
peer 4.4.4.4 enable
peer 192.168.0.1 enable
#
ospf 1
area 0.0.0.0
network 2.2.2.2 0.0.0.0
network 10.1.1.0 0.0.0.255
network 10.1.3.0 0.0.0.255
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
#
interface LoopBack0
ip address 3.3.3.3 255.255.255.255
#
bgp 65009
router-id 3.3.3.3
peer 2.2.2.2 as-number 65009
peer 2.2.2.2 connect-interface LoopBack0
peer 4.4.4.4 as-number 65009
peer 4.4.4.4 connect-interface LoopBack0
#
ipv4-family unicast
undo synchronization
peer 2.2.2.2 enable
peer 4.4.4.4 enable
#
ospf 1
area 0.0.0.0
network 3.3.3.3 0.0.0.0
network 10.1.2.0 0.0.0.255
network 10.1.3.0 0.0.0.255
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
interface LoopBack0
ip address 4.4.4.4 255.255.255.255
#
```

```
bgp 65009
router-id 4.4.4.4
peer 2.2.2.2 as-number 65009
peer 2.2.2.2 connect-interface LoopBack0
peer 3.3.3.3 as-number 65009
peer 3.3.3.3 connect-interface LoopBack0
#
ipv4-family unicast
undo synchronization
peer 2.2.2.2 enable
peer 3.3.3.3 enable
#
ospf 1
area 0.0.0.0
network 4.4.4.4 0.0.0.0
network 10.1.1.0 0.0.0.255
network 10.1.2.0 0.0.0.255
#
return
```

Example for Configuring BGP to Interact with an IGP

Configuring BGP to interact with an IGP can enrich routing tables.

Networking Requirements

As the Internet grows, devices in different networks need to access each other, data needs to be reliably transmitted, and the traffic interruption time needs to be minimized. This requires that routing information be transmitted widely and network convergence be accelerated. BGP can transmit routing information efficiently and widely. BGP, however, does not calculate routes by itself. An IGP can implement rapid route convergence, but it transmits routing information with a low efficiency in a limited scope. After BGP is configured to interact with an IGP, IGP routes can be imported into the BGP routing table and transmitted efficiently. BGP routes can also be imported into the IGP routing table to allow access to other ASs.

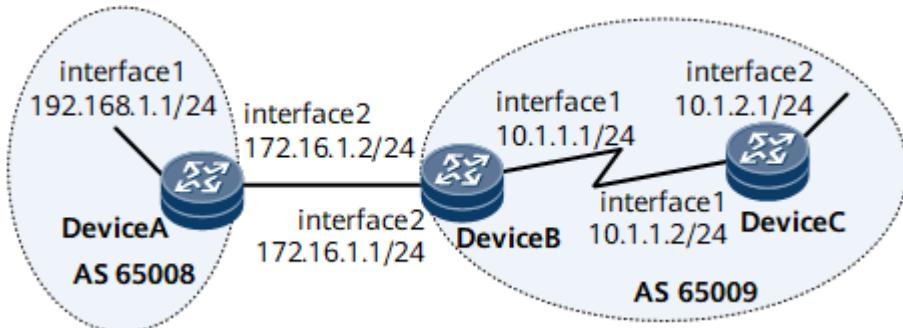
The network shown in [Figure 1-426](#) is divided into AS 65008 and AS 65009. In AS 65009, an IGP is used to calculate routes. In this example, OSPF is used as the IGP. BGP can be configured to enable the two ASs to access each other. Interaction between BGP and the IGP can be configured on edge routers of the two ASs so that the two ASs can exchange routes efficiently. In addition, AS external routes can be imported into the IGP routing table to allow access to the outside of the local AS.

Figure 1-426 Configuring BGP to interact with an IGP



NOTE

Interfaces 1 and 2 in this example represent GE 1/0/0 and GE 2/0/0, respectively.



Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure OSPF on Device B and Device C.
2. Establish an EBGP connection between Device A and Device B.
3. Configure BGP and OSPF to import routes from each other on Device B and then check the routes.
4. Configure BGP route summarization on Device B to simplify the BGP routing table.

Data Preparation

To complete the configuration, you need the following data:

- Router ID and AS number of Device A
- Router IDs and AS numbers of Device B and Device C

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure OSPF.

Configure Device B.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
[~DeviceB-ospf-1] quit
```

Configure Device C.

```
[~DeviceC] ospf 1
```

```
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] network 10.1.2.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
[~DeviceC-ospf-1] quit
```

Step 3 Configure an EBGP connection.

Configure Device A.

```
[~DeviceA] bgp 65008
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 172.16.1.1 as-number 65009
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] network 192.168.1.0 255.255.255.0
[*DeviceA-bgp-af-ipv4] commit
```

Configure Device B.

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 172.16.1.2 as-number 65008
[*DeviceB-bgp] commit
```

Step 4 Configure BGP to interact with an IGP.

Configure BGP to import OSPF routes on Device B.

```
[~DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] import-route ospf 1
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
[~DeviceB-bgp] quit
```

Check the routing table of Device A.

```
[~DeviceA] display bgp routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 3
Network          NextHop        MED      LocPrf    PrefVal Path/Ogn
* 192.168.1.0/24 0.0.0.0      0          0          i
* 10.1.1.0/24    172.16.1.1   1          0          65009?
* 10.1.2.0/24    172.16.1.1   2          0          65009?
```

Configure OSPF to import BGP routes on Device B.

```
[~DeviceB] ospf
[*DeviceB-ospf-1] import-route bgp
[*DeviceB-ospf-1] commit
[~DeviceB-ospf-1] quit
```

Check the routing table of Device C.

```
[~DeviceC] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table : _public_
Destinations : 12      Routes : 12
Destination/Mask Proto Pre Cost      Flags NextHop      Interface
192.168.1.0/24  O_ASE 150 1        D 10.1.1.1      GigabitEthernet1/0/0
```

10.1.1.0/24	Direct	0	0	D	10.1.1.2	GigabitEthernet1/0/0
10.1.1.1/32	Direct	0	0	D	10.1.1.1	GigabitEthernet1/0/0
10.1.1.2/32	Direct	0	0	D	127.0.0.1	GigabitEthernet1/0/0
10.1.1.255/32	Direct	0	0	D	127.0.0.1	GigabitEthernet1/0/0
10.1.2.0/24	Direct	0	0	D	10.1.2.1	GigabitEthernet2/0/0
10.1.2.1/32	Direct	0	0	D	127.0.0.1	GigabitEthernet2/0/0
10.1.2.255/32	Direct	0	0	D	127.0.0.1	GigabitEthernet2/0/0
127.0.0.0/8	Direct	0	0	D	127.0.0.1	InLoopBack0
127.0.0.1/32	Direct	0	0	D	127.0.0.1	InLoopBack0
127.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0
255.255.255.255/32	Direct	0	0	D	127.0.0.1	InLoopBack0

Step 5 Configure automatic route summarization.

Configure Device B.

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] summary automatic
[*DeviceB-bgp-af-ipv4] commit
```

Check the BGP routing table of Device A.

```
[~DeviceA] display bgp routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
               h - history, i - internal, s - suppressed, S - Stale
               Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
*> 192.168.1.0/24 0.0.0.0      0        0       i
*> 10.0.0.0       172.16.1.1   0        0       65009?
```

Verify the configuration by using the ping command.

```
[~DeviceA] ping -a 192.168.1.1 10.1.2.1
PING 10.1.2.1: 56 data bytes, press CTRL_C to break
Reply from 10.1.2.1: bytes=56 Sequence=1 ttl=254 time=15 ms
Reply from 10.1.2.1: bytes=56 Sequence=2 ttl=254 time=31 ms
Reply from 10.1.2.1: bytes=56 Sequence=3 ttl=254 time=47 ms
Reply from 10.1.2.1: bytes=56 Sequence=4 ttl=254 time=46 ms
Reply from 10.1.2.1: bytes=56 Sequence=5 ttl=254 time=47 ms
--- 10.1.2.1 ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 15/37/47 ms
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.2 255.255.255.0
```

```
#  
bgp 65008  
router-id 1.1.1.1  
peer 172.16.1.1 as-number 65009  
#  
ipv4-family unicast  
undo synchronization  
network 192.168.1.0 255.255.255.0  
peer 172.16.1.1 enable  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.1.1 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 172.16.1.1 255.255.255.0  
#  
bgp 65009  
router-id 2.2.2.2  
peer 172.16.1.2 as-number 65008  
#  
ipv4-family unicast  
undo synchronization  
summary automatic  
import-route ospf 1  
peer 172.16.1.2 enable  
#  
ospf 1  
import-route bgp  
area 0.0.0  
network 10.1.1.0 0.0.0.255  
#  
return
```

- Device C configuration file

```
#  
sysname DeviceC  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.1.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.2.1 255.255.255.0  
#  
ospf 1  
area 0.0.0  
network 10.1.1.0 0.0.0.255  
network 10.1.2.0 0.0.0.255  
#  
return
```

Example for Configuring the MED Attribute to Control Route Selection

By setting the MED attribute, you can flexibly control BGP route selection.

Networking Requirements

The MED attribute equals a metric used in an IGP, and is used to determine the optimal route for traffic that enters an AS. When a BGP device obtains multiple

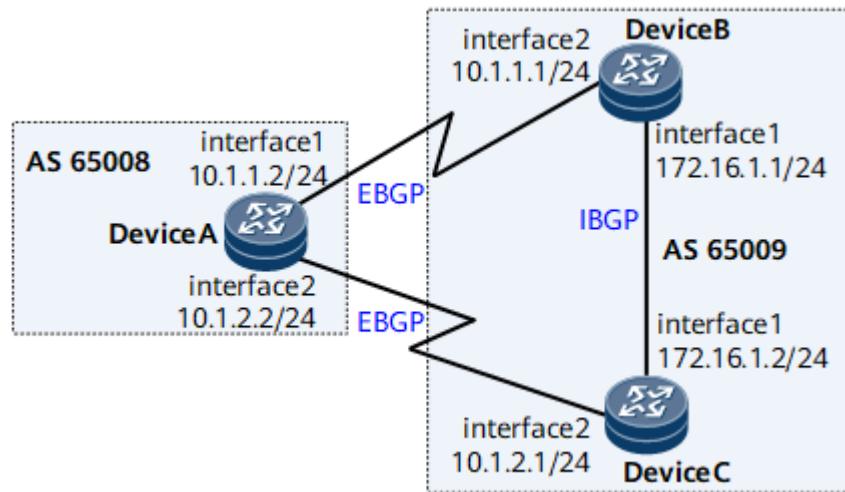
routes to the same destination address but with different next hops from EBGP peers, the route with the smallest MED value is selected as the optimal route.

On the network shown in [Figure 1-427](#), BGP is configured on all routers, routerA is in AS 65008, and routerB and routerC are in AS 65009. In addition, routerA establishes EBGP connections with routerB and routerC, whereas routerB establishes an IBGP connection with routerC. Traffic sent by routerA to 172.16.1.0 can enter AS 65009 through routerB or routerC. If other conditions are the same, you can configure routerB or routerC to change the MED value of the route advertised to routerA to select the ingress for traffic to enter AS 65009.

Figure 1-427 Configuring the MED attribute to control route selection

 NOTE

Interfaces 1 through 2 in this example are GE 1/0/0 and GE 2/0/0, respectively.



Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

Configuration Roadmap

The configuration roadmap is as follows:

1. Establish EBGP connections between Device A and Device B, and between Device A and Device C, and establish an IBGP connection between Device B and Device C.
2. Apply a routing policy to increase the MED value of the route sent by Device B to Device A so that Device A will send traffic to AS 65009 through Device C.

Data Preparation

To complete the configuration, you need the following data:

- Router ID 1.1.1.1 and AS number 65008 of Device A
- Router IDs 2.2.2.2 and 3.3.3.3, and AS number 65009 of Devices B and C
- New MED value 100 of the route on Device B

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

Step 2 Configure BGP connections.

Configure Device A.

```
[~DeviceA] bgp 65008
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.1.1.1 as-number 65009
[*DeviceA-bgp] peer 10.1.2.1 as-number 65009
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

Configure Device B.

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.1.1.2 as-number 65008
[*DeviceB-bgp] peer 172.16.1.2 as-number 65009
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] network 172.16.1.0 255.255.255.0
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
[~DeviceB-bgp] quit
```

Configure Device C.

```
[~DeviceC] bgp 65009
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.1.2.2 as-number 65008
[*DeviceC-bgp] peer 172.16.1.1 as-number 65009
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] network 172.16.1.0 255.255.255.0
[*DeviceC-bgp-af-ipv4] commit
[~DeviceC-bgp-af-ipv4] quit
[~DeviceC-bgp] quit
```

Check the routing table of Device A.

```
[~DeviceA] display bgp routing-table 172.16.1.0 24

BGP local router ID : 1.1.1.1
Local AS number : 65008
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.1.1 (2.2.2.2)
Route Duration: 0d00h00m56s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 65009, origin igp, MED 0, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
    10.1.1.1
    10.1.2.1

BGP routing table entry information of 172.16.1.0/24:
From: 10.1.2.1 (3.3.3.3)
Route Duration: 0d00h00m06s
Direct Out-interface: GigabitEthernet2/0/0
```

```
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 65009, origin igp, MED 0, pref-val 0, valid, external, pre 255, not preferred for router ID
Not advertised to any peers yet
```

The command output shows that there are two valid routes to 172.16.1.0/24. The route with 10.1.1.1 as the next hop is the optimal route because the router ID of Device B is smaller.

Step 3 Configure the MED attribute.

```
# Set the MED value for the route sent by Device B to Device A based on a routing policy.
```

```
[~DeviceB] route-policy 10 permit node 10
[*DeviceB-route-policy] apply cost 100
[*DeviceB-route-policy] commit
[~DeviceB-route-policy] quit
[~DeviceB] bgp 65009
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] peer 10.1.1.2 route-policy 10 export
[~DeviceB-bgp-af-ipv4] commit
```

```
# Check the routing table of Device A.
```

```
[~DeviceA] display bgp routing-table 172.16.1.0 24

BGP local router ID : 1.1.1.1
Local AS number : 65008
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.2.1 (3.3.3.3)
Route Duration: 0d00h07m45s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 65009, origin igp, MED 0, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
    10.1.1.1
    10.1.2.1

BGP routing table entry information of 172.16.1.0/24:
From: 10.1.1.1 (2.2.2.2)
Route Duration: 0d00h00m08s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 65009, origin igp, MED 100, pref-val 0, valid, external, pre 255, not preferred for MED
Not advertised to any peers yet
```

The command output shows that the MED value of the next hop 10.1.1.1 (Device B) is 100 and that the MED value of the next hop 10.1.2.1 is 0. Therefore, the route with the smaller MED value is selected.

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
```

```
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
bgp 65008
router-id 1.1.1.1
peer 10.1.1.1 as-number 65009
peer 10.1.2.1 as-number 65009
#
ipv4-family unicast
peer 10.1.1.1 enable
peer 10.1.2.1 enable
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
#
bgp 65009
router-id 2.2.2.2
peer 172.16.1.2 as-number 65009
peer 10.1.1.2 as-number 65008
#
ipv4-family unicast
undo synchronization
network 172.16.1.0 255.255.255.0
peer 172.16.1.2 enable
peer 10.1.1.2 enable
peer 10.1.1.2 route-policy 10 export
#
route-policy 10 permit node 10
apply cost 100
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
#
bgp 65009
router-id 3.3.3.3
peer 172.16.1.1 as-number 65009
peer 10.1.2.2 as-number 65008
#
ipv4-family unicast
undo synchronization
network 172.16.1.0 255.255.255.0
peer 172.16.1.1 enable
peer 10.1.2.2 enable
#
return
```

Example for Configuring an AS_Path Filter

AS_Path filters can be used to improve network performance.

Networking Requirements

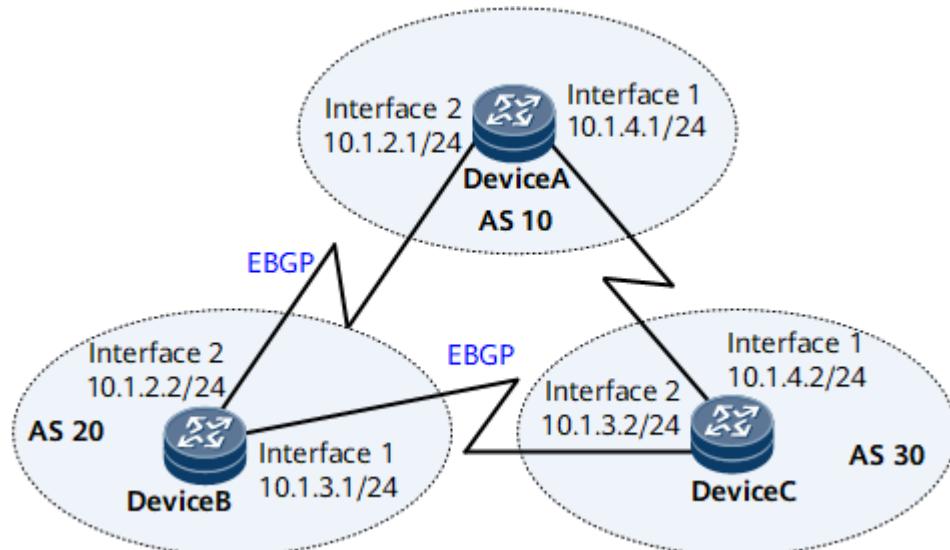
Enterprises A, B, and C belong to different ASs. The network of enterprise B communicates with the networks of the other two enterprises through EBGP. Due to the competition relationship, enterprises A and C require that the routes that they advertise to enterprise B be not learned by each other. In this situation, configure an AS_Path filter on enterprise B.

In [Figure 1-428](#), Device B establish EBGP connections with Devices A and C. To disable devices in AS 10 from communicating with devices in AS 30, you can configure an AS_Path filter on Device B to prevent devices in AS 20 from advertising routes of AS 30 to AS 10 or routes of AS 10 to AS 30.

Figure 1-428 Configuring BGP route filtering

 NOTE

Interfaces 1 through 2 in this example are GE 1/0/0 and GE 2/0/0, respectively.



Precautions

During the configuration process, note the following:

- The relationship between multiple filtering rules in the same filter is OR.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

Configuration Roadmap

The configuration roadmap is as follows:

1. Establish EBGP connections between Device A and Device B, and between Device B and Device C, and then import direct routes.

2. Configure an AS_Path filter on Device B and then apply its filtering rules.

Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A, Device B, and Device C
- Number of the AS_Path filter

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure EBGP connections.

Configure Device A.

```
[~DeviceA] bgp 10
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.1.2.2 as-number 20
[*DeviceA-bgp] import-route direct
[*DeviceA-bgp] commit
```

Configure Device B.

```
[*DeviceB] bgp 20
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.1.2.1 as-number 10
[*DeviceB-bgp] peer 10.1.3.2 as-number 30
[*DeviceB-bgp] import-route direct
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

Configure Device C.

```
[~DeviceC] bgp 30
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.1.3.1 as-number 20
[*DeviceC-bgp] import-route direct
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

Display the routing table advertised by Device B. Use the routes advertised by Device B to Device C as an example. You can view that Device B advertises the routes destined for the network segment between Device A and Device C.

```
<DeviceB> display bgp routing-table peer 10.1.3.2 advertised-routes
```

```
BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 3					
	Network	NextHop	MED	LocPrf	PrefVal Path/Ogn
*>	10.1.2.0/24	10.1.3.1	0	0	20?
*>	10.1.3.0/24	10.1.3.1	0	0	20?
*>	10.1.4.0/24	10.1.3.1	0	0	20 10?

Check the routing table of Device C. The command output shows that Device C has learned the two routes advertised by Device B.

```
<DeviceC> display bgp routing-table

BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 7
Network          NextHop          MED      LocPrf  PrefVal Path/Ogn
*> 10.1.2.0/24    10.1.3.1        0         0      20?
*> 10.1.3.0/24    0.0.0.0        0         0      ?
                  10.1.3.1        0         0      20?
*> 10.1.3.2/32    0.0.0.0        0         0      ?
*> 10.1.4.0/24    0.0.0.0        0         0      ?
*          10.1.3.1        0         0      20 10?
*> 10.1.4.2/32    0.0.0.0        0         0      ?
```

Step 3 Configure the AS_Path filter on Device B and then apply the filter on the outbound interface of Device B.

```
# Create AS_Path filter 1 to deny the routes carrying AS 30. The regular expression
"_30_" indicates any AS list that contains AS 30 and ".*" matches any character.
```

```
[~DeviceB] ip as-path-filter 1 deny _30_
[*DeviceB] ip as-path-filter 1 permit .*
[*DeviceB] commit
```

```
# Create AS_Path filter 2 to deny the routes carrying AS 10. The regular expression
"_10_" indicates any AS list that contains AS 10 and ".*" matches any character.
```

```
[~DeviceB] ip as-path-filter 2 deny _10_
[*DeviceB] ip as-path-filter 2 permit .*
[*DeviceB] commit
```

```
# Apply the AS_Path filter on two outbound interfaces of Device B.
```

```
[~DeviceB] bgp 20
[*DeviceB-bgp] peer 10.1.2.1 as-path-filter 1 export
[*DeviceB-bgp] peer 10.1.3.2 as-path-filter 2 export
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

Step 4 Verify the configuration.

```
# Display the routing table advertised by Device B. The command output shows
that the advertised routes to the network segment between Device A and Device
C do not exist. Use the routes advertised by Device B to Device C as an example.
```

```
<DeviceB> display bgp routing-table peer 10.1.3.2 advertised-routes

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
Network          NextHop          MED      LocPrf  PrefVal Path/Ogn
*> 10.1.2.0/24    10.1.3.1        0         0      20?
*> 10.1.3.0/24    10.1.3.1        0         0      20?
```

Similarly, the BGP routing table of Device C does not have these routes.

```
<DeviceC> display bgp routing-table

BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
Network          NextHop          MED      LocPrf  PrefVal Path/Ogn
*> 10.1.2.0/24    10.1.3.1        0         0       20?
*> 10.1.3.0/24    0.0.0.0        0         0       ?
              10.1.3.1        0         0       20?
*> 10.1.3.2/32    0.0.0.0        0         0       ?
*> 10.1.4.0/24    0.0.0.0        0         0       ?
*> 10.1.4.2/32    0.0.0.0        0         0       ?
```

Check the routing table advertised by Device B, and you can view that the advertised routes to the network segment between Device A and Device C do not exist. Use the routes advertised by Device B to Device A as an example.

```
<DeviceB> display bgp routing-table peer 10.1.2.1 advertised-routes

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

```
Total Number of Routes: 2
Network          NextHop          MED      LocPrf  PrefVal Path/Ogn
*> 10.1.2.0/24    10.1.2.2        0         0       20?
*> 10.1.3.0/24    10.1.2.2        0         0       20?
```

Similarly, the BGP routing table of Device A does not have these routes.

```
<DeviceA> display bgp routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
Network          NextHop          MED      LocPrf  PrefVal Path/Ogn
*> 10.1.2.0/24    0.0.0.0        0         0       ?
              10.1.2.2        0         0       20?
*> 10.1.2.1/32    0.0.0.0        0         0       ?
*> 10.1.3.0/24    10.1.2.2        0         0       20?
*> 10.1.4.0/24    0.0.0.0        0         0       ?
*> 10.1.4.1/32    0.0.0.0        0         0       ?
```

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA
```

```
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.4.1 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.2.1 255.255.255.0  
#  
bgp 10  
router-id 1.1.1.1  
peer 10.1.2.2 as-number 20  
#  
ipv4-family unicast  
undo synchronization  
import-route direct  
peer 10.1.2.2 enable  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.3.1 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.2.2 255.255.255.0  
#  
bgp 20  
router-id 2.2.2.2  
peer 10.1.2.1 as-number 10  
peer 10.1.3.2 as-number 30  
#  
ipv4-family unicast  
undo synchronization  
import-route direct  
peer 10.1.2.1 enable  
peer 10.1.2.1 as-path-filter 1 export  
peer 10.1.3.2 enable  
peer 10.1.3.2 as-path-filter 2 export  
#  
ip as-path-filter 1 deny _30_  
ip as-path-filter 1 permit .*  
ip as-path-filter 2 deny _10_  
ip as-path-filter 2 permit .*  
#  
return
```

- Device C configuration file

```
#  
sysname DeviceC  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.4.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.3.2 255.255.255.0  
#  
bgp 30  
router-id 3.3.3.3  
peer 10.1.3.1 as-number 20  
#  
ipv4-family unicast
```

```
undo synchronization
import-route direct
peer 10.1.3.1 enable
#
return
```

Example for Configuring BGP RRs

With BGP RRs, IBGP peers do not have to be fully meshed, which reduces configuration workload and facilitates network maintenance.

Networking Requirements

On a large-scale network, multiple routers that run BGP are deployed within an AS. These routers need to use BGP to advertise routes to each other. To meet this need, IBGP peer relationships need to be set up between all the routers. However, fully meshed connections between all routers increase router costs and the configuration workload and are difficult to maintain. A solution that can simplify network configuration and reduce router costs without affecting route transmission is required.

To address this issue, configure RRs. In **Figure 1-429**, AS 65010 is divided into two clusters: Cluster 1 and Cluster 2. DeviceB is configured as an RR in Cluster 1, and DeviceD and DeviceE are its clients. DeviceC is configured as an RR in Cluster 2, and DeviceF, DeviceG, and DeviceH are its clients. DeviceA is the non-client of DeviceB and DeviceC. DeviceB and DeviceC are non-clients of each other.

Figure 1-429 Configuring BGP RRs



Interfaces 1 through 5 in this example represent GE 1/0/0, GE 2/0/0, GE 3/0/0, GE 1/0/1, and GE 1/0/2, respectively.

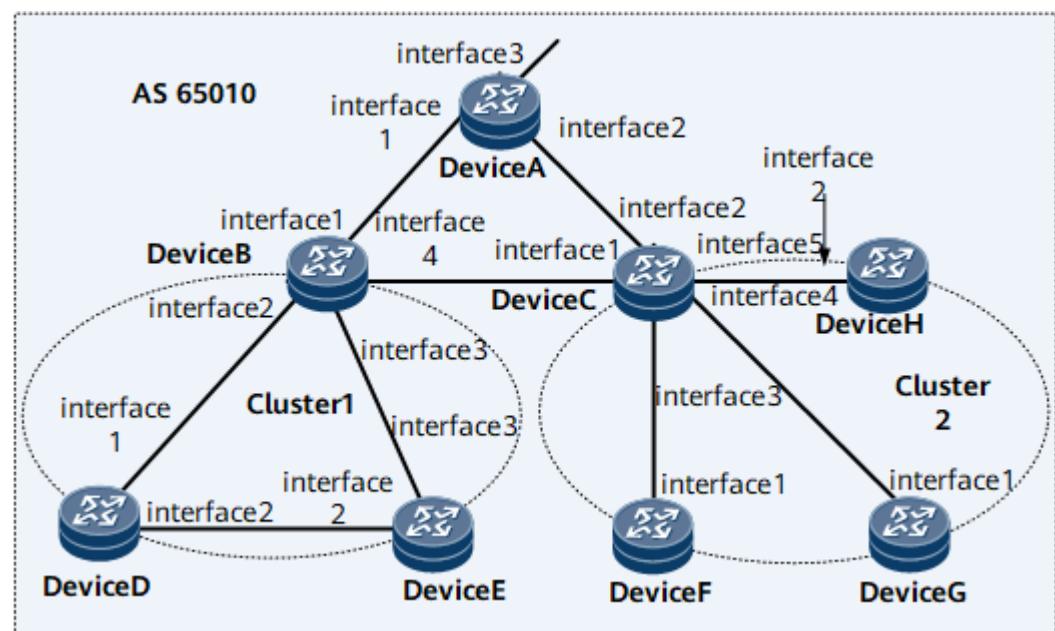


Table 1-154 IP addresses of the interfaces

Device	Interface	IP Address	Device	Interface	IP Address
Device A	GE 3/0/0	172.16.1.1/24	Device C	GE 1/0/1	10.1.8.1/24
	GE 1/0/0	10.1.1.2/24		GE 1/0/2	10.1.9.1/24
	GE 2/0/0	10.1.3.2/24	Device D	GE 1/0/0	10.1.4.2/24
Device B	GE 1/0/0	10.1.1.1/24		GE 2/0/0	10.1.6.1/24
	GE 2/0/0	10.1.4.1/24	Device E	GE 2/0/0	10.1.6.2/24
	GE 3/0/0	10.1.5.1/24		GE 3/0/0	10.1.5.2/24
	GE 1/0/1	10.1.2.1/24	Device F	GE 1/0/0	10.1.7.2/24
Device C	GE 1/0/0	10.1.2.2/24	Device G	GE 1/0/0	10.1.8.2/24
	GE 2/0/0	10.1.3.1/24	Device H	GE 2/0/0	10.1.9.2/24
	GE 3/0/0	10.1.7.1/24			

Precautions

When configuring a BGP RR, note the following rules:

- If a cluster has multiple RRs, run the **reflector cluster-id** command to set the same cluster ID for these RRs to prevent routing loops.
- The name of a peer group is case sensitive.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IBGP connections between the clients and the RR, and between the non-client and the RR.
2. Configure Device B and Device C as RRs, specify their clients, and check routes.

Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A, Device B, Device C, Device D, Device E, Device F, Device G, and Device H
- Cluster ID of Device B

Procedure

- Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure IBGP connections between the clients and the RR, and between the non-client and the RR.
- Step 3** Configure RRs.

Configure Device B.

```
[~DeviceB] bgp 65010
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] group in_rr internal
[*DeviceB-bgp] peer 10.1.4.2 group in_rr
[*DeviceB-bgp] peer 10.1.5.2 group in_rr
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] peer in_rr reflect-client
[*DeviceB-bgp-af-ipv4] undo reflect between-clients
[*DeviceB-bgp-af-ipv4] reflector cluster-id 1
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
```

Configure Device C.

```
[~DeviceC] bgp 65010
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] group in_rr internal
[*DeviceC-bgp] peer 10.1.7.2 group in_rr
[*DeviceC-bgp] peer 10.1.8.2 group in_rr
[*DeviceC-bgp] peer 10.1.9.2 group in_rr
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] peer in_rr reflect-client
[*DeviceC-bgp-af-ipv4] reflector cluster-id 2
[*DeviceC-bgp-af-ipv4] commit
[~DeviceC-bgp-af-ipv4] quit
```

Check the routing table of Device D.

```
[~DeviceD] display bgp routing-table 172.16.1.0

BGP local router ID : 4.4.4.4
Local AS number : 65010
Paths: 1 available, 0 best, 0 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.4.1 (2.2.2.2)
Route Duration: 00h00m14s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface:
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path Nil, origin igp, MED 0, localpref 100, pref-val 0, internal, pre 255
Originator: 1.1.1.1
Cluster list: 0.0.0.1
Not advertised to any peer yet
```

The command output shows that Device D has learned from Device B the route advertised by Device A and that the Originator and Cluster_ID attributes of this route are displayed.

----End

Configuration Files

- Device A configuration file

```
#  
sysname DeviceA  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.1.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.3.2 255.255.255.0  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 172.16.1.1 255.255.255.0  
#  
bgp 65010  
router-id 1.1.1.1  
peer 10.1.1.1 as-number 65010  
peer 10.1.3.1 as-number 65010  
#  
ipv4-family unicast  
undo synchronization  
network 172.16.1.0 255.255.255.0  
peer 10.1.1.1 enable  
peer 10.1.3.1 enable  
#  
return
```

- Device B configuration file

```
#  
sysname DeviceB  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 10.1.1.1 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 10.1.4.1 255.255.255.0  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 10.1.5.1 255.255.255.0  
#  
interface GigabitEthernet1/0/1  
undo shutdown  
ip address 10.1.2.1 255.255.255.0  
#  
bgp 65010  
router-id 2.2.2.2  
peer 10.1.1.2 as-number 65010  
peer 10.1.2.2 as-number 65010  
group in_rr internal  
peer 10.1.4.2 as-number 65010  
peer 10.1.4.2 group in_rr  
peer 10.1.5.2 as-number 65010  
peer 10.1.5.2 group in_rr  
#  
ipv4-family unicast  
undo synchronization  
undo reflect between-clients  
reflector cluster-id 1  
peer 10.1.1.2 enable  
peer 10.1.2.2 enable  
peer in_rr enable  
peer in_rr reflect-client  
peer 10.1.4.2 enable  
peer 10.1.4.2 group in_rr  
peer 10.1.5.2 enable  
peer 10.1.5.2 group in_rr
```

- ```

return
● Device C configuration file

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.7.1 255.255.255.0

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.8.1 255.255.255.0

interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.9.1 255.255.255.0

bgp 65010
router-id 3.3.3.3
peer 10.1.2.1 as-number 65010
peer 10.1.3.2 as-number 65010
group in_rr internal
peer 10.1.7.2 as-number 65010
peer 10.1.7.2 group in_rr
peer 10.1.8.2 as-number 65010
peer 10.1.8.2 group in_rr
peer 10.1.9.2 as-number 65010
peer 10.1.9.2 group in_rr

ipv4-family unicast
undo synchronization
reflector cluster-id 2
peer 10.1.2.1 enable
peer 10.1.3.2 enable
peer in_rr enable
peer in_rr reflect-client
peer 10.1.7.2 enable
peer 10.1.7.2 group in_rr
peer 10.1.8.2 enable
peer 10.1.8.2 group in_rr
peer 10.1.9.2 enable
peer 10.1.9.2 group in_rr

return
```

- Device D configuration file

```

sysname DeviceD

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.6.1 255.255.255.0

bgp 65010
router-id 4.4.4.4
peer 10.1.4.1 as-number 65010
```

```

ipv4-family unicast
undo synchronization
peer 10.1.4.1 enable

return
```

 NOTE

Configuration files of other routers are similar to the Device D configuration file.

## Example for Configuring a BGP Confederation

BGP confederations can be used to reduce the number of IBGP connections.

## Networking Requirements

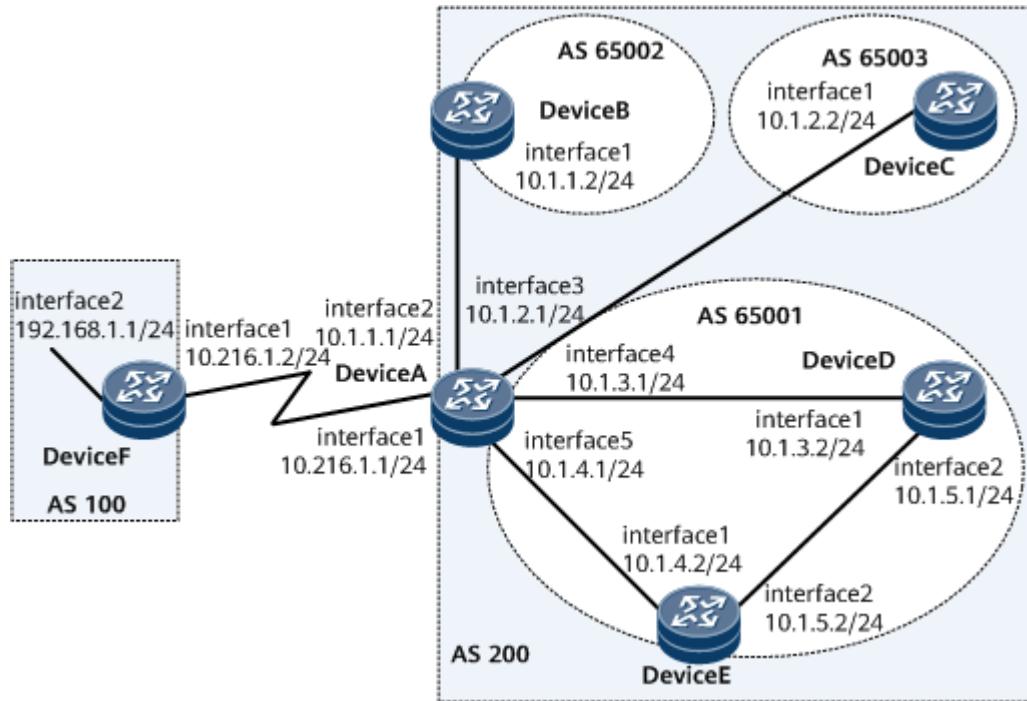
If multiple devices are deployed in an AS and fully meshed IBGP connections must be implemented between every two devices in the AS, a large number of IBGP connections will be established, increasing operation and maintenance costs. To address this issue, configure BGP confederations.

In [Figure 1-430](#), to implement interworking between devices in AS 200, full-mesh IBGP connections need to be established between the devices. However, because multiple routers run BGP in AS 200, the cost for establishing a fully meshed network is high. To reduce the number of IBGP connections to be established, you can configure the confederation function on the devices in AS 200. The confederation solution can deal with the increase of IBGP connections in an AS. As shown in [Figure 1-430](#), AS 200 is divided into three sub-ASs: AS 65001, AS 65002, and AS 65003. AS 65001 establishes confederation EBGP peer relationships with AS 65002 and AS 65003. The three routers in AS 65001 establish IBGP fully meshed connections. This greatly reduces the number of IBGP connections to be established in AS 200 and reduces O&M costs.

**Figure 1-430** Configuring the confederation

 NOTE

Interfaces 1 through 5 in this example represent GE1/0/0, GE2/0/0, GE3/0/0, GE1/0/1, and GE1/0/2, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure a BGP confederation on each router in AS 200.
2. Configure the IBGP connection in AS 65001.
3. Configure the EBGP connection between AS 100 and AS 200, and check the routes.

## Data Preparation

To complete the configuration, you need the following data:

- The Router IDs of DeviceA, DeviceB, DeviceC, DeviceD, DeviceE, and DeviceF (1.1.1.1, 2.2.2.2, 3.3.3.3, 4.4.4.4, 5.5.5.5, and 6.6.6.6)
- The AS number (100), and the three sub-ASs of AS 200 (AS 65001, AS 65002, and AS 65003)

## Procedure

### Step 1 Assign an IP address to each interface.

For configuration details, see [Configuration Files](#) in this section.

### Step 2 Configure the BGP confederation.

# Configure DeviceA.

```
[~DeviceA] bgp 65001
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] confederation id 200
[*DeviceA-bgp] confederation peer-as 65002 65003
[*DeviceA-bgp] peer 10.1.1.2 as-number 65002
[*DeviceA-bgp] peer 10.1.2.2 as-number 65003
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 10.1.1.2 next-hop-local
[*DeviceA-bgp-af-ipv4] peer 10.1.2.2 next-hop-local
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~DeviceB] bgp 65002
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] confederation id 200
[*DeviceB-bgp] confederation peer-as 65001
[*DeviceB-bgp] peer 10.1.1.1 as-number 65001
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure DeviceC.

```
[~DeviceC] bgp 65003
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] confederation id 200
[*DeviceC-bgp] confederation peer-as 65001
[*DeviceC-bgp] peer 10.1.2.1 as-number 65001
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

### Step 3 Configure IBGP connections inside AS 65001.

# Configure DeviceA.

```
[~DeviceA] bgp 65001
[*DeviceA-bgp] peer 10.1.3.2 as-number 65001
[*DeviceA-bgp] peer 10.1.4.2 as-number 65001
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 10.1.3.2 next-hop-local
[*DeviceA-bgp-af-ipv4] peer 10.1.4.2 next-hop-local
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

# Configure DeviceD.

```
[~DeviceD] bgp 65001
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] confederation id 200
[*DeviceD-bgp] peer 10.1.3.1 as-number 65001
[*DeviceD-bgp] peer 10.1.5.2 as-number 65001
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

# Configure DeviceE.

```
[~DeviceE] bgp 65001
[*DeviceE-bgp] router-id 5.5.5.5
[*DeviceE-bgp] confederation id 200
[*DeviceE-bgp] peer 10.1.4.1 as-number 65001
[*DeviceE-bgp] peer 10.1.5.1 as-number 65001
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

**Step 4** Configure the EBGP connection between AS 100 and AS 200.

# Configure DeviceA.

```
[~DeviceA] bgp 65001
[*DeviceA-bgp] peer 10.216.1.2 as-number 100
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure DeviceF.

```
[~DeviceF] bgp 100
[*DeviceF-bgp] router-id 6.6.6.6
[*DeviceF-bgp] peer 10.216.1.1 as-number 200
[*DeviceF-bgp] ipv4-family unicast
[*DeviceF-bgp-af-ipv4] network 192.168.1.0 255.255.255.0
[*DeviceF-bgp-af-ipv4] commit
[~DeviceF-bgp-af-ipv4] quit
[~DeviceF-bgp] quit
```

**Step 5** Verify the configuration.

# Check the BGP routing table of DeviceB.

```
[~DeviceB] display bgp routing-table

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
 Network NextHop MED LocPrf PrefVal Path/Ogn
*>i 192.168.1.0/24 10.1.1.1 0 100 0 (65001) 100i

[~DeviceB] display bgp routing-table 192.168.1.0

BGP local router ID : 2.2.2.2
Local AS number : 65002
Paths: 1 available, 1 best, 1 select
BGP routing table entry information of 192.168.1.0/24:
From: 10.1.1.1 (1.1.1.1)
Route Duration: 00h12m29s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path (65001) 100, origin igp, MED 0, localpref 100, pref-val 0, valid, external-confed, best, select,
active, pre 255
Not advertised to any peer yet
```

# Check the BGP routing table of DeviceD.

```
[~DeviceD] display bgp routing-table

BGP Local router ID is 4.4.4.4
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
 Network NextHop MED LocPrf PrefVal Path/Ogn
*>i 192.168.1.0/24 10.1.3.1 0 100 0 100i

[~DeviceD] display bgp routing-table 192.168.1.0

BGP local router ID : 4.4.4.4
Local AS number : 65001
```

```
Paths: 1 available, 1 best, 1 select
BGP routing table entry information of 192.168.1.0/24:
From: 10.1.3.1 (1.1.1.1)
Route Duration: 00h23m57s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-Interface: GigabitEthernet1/0/0
Original nexthop: 10.1.3.1
Qos information : 0x0
AS-path 100, origin igrp, MED 0, localpref 100, pref-val 0, valid, internal-confed, best, select, active, pre 255
Not advertised to any peer yet
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.216.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.3.1 255.255.255.0

interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.4.1 255.255.255.0

bgp 65001
router-id 1.1.1.1
confederation id 200
confederation peer-as 65002 65003
peer 10.1.1.2 as-number 65002
peer 10.1.2.2 as-number 65003
peer 10.1.3.2 as-number 65001
peer 10.1.4.2 as-number 65001
peer 10.216.1.2 as-number 100

ipv4-family unicast
undo synchronization
peer 10.216.1.2 enable
peer 10.1.1.2 enable
peer 10.1.1.2 next-hop-local
peer 10.1.2.2 enable
peer 10.1.2.2 next-hop-local
peer 10.1.3.2 enable
peer 10.1.3.2 next-hop-local
peer 10.1.4.2 enable
peer 10.1.4.2 next-hop-local

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
```

```
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
bgp 65002
router-id 2.2.2.2
confederation id 200
confederation peer-as 65001
peer 10.1.1.1 as-number 65001
#
ipv4-family unicast
undo synchronization
peer 10.1.1.1 enable
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
bgp 65003
router-id 3.3.3.3
confederation id 200
confederation peer-as 65001
peer 10.1.2.1 as-number 65001
#
ipv4-family unicast
undo synchronization
peer 10.1.2.1 enable
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.5.1 255.255.255.0
#
bgp 65001
router-id 4.4.4.4
confederation id 200
peer 10.1.3.1 as-number 65001
peer 10.1.5.2 as-number 65001
#
ipv4-family unicast
undo synchronization
peer 10.1.3.1 enable
peer 10.1.5.2 enable
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
```

```
ip address 10.1.5.2 255.255.255.0
#
bgp 65001
router-id 5.5.5.5
confederation id 200
peer 10.1.4.1 as-number 65001
peer 10.1.5.1 as-number 65001
#
ipv4-family unicast
undo synchronization
peer 10.1.4.1 enable
peer 10.1.5.1 enable
#
return
```

#### NOTE

The configuration file of DeviceE is similar to that of DeviceD.

- **DeviceF configuration file**

```
#
sysname DeviceF
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.216.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
bgp 100
router-id 6.6.6.6
peer 10.216.1.1 as-number 200
#
ipv4-family unicast
undo synchronization
network 192.168.1.0 255.255.255.0
peer 10.216.1.1 enable
#
return
```

## Example for Configuring the BGP Community Attribute

By setting the community attribute, you can flexibly control BGP route selection.

## Networking Requirements

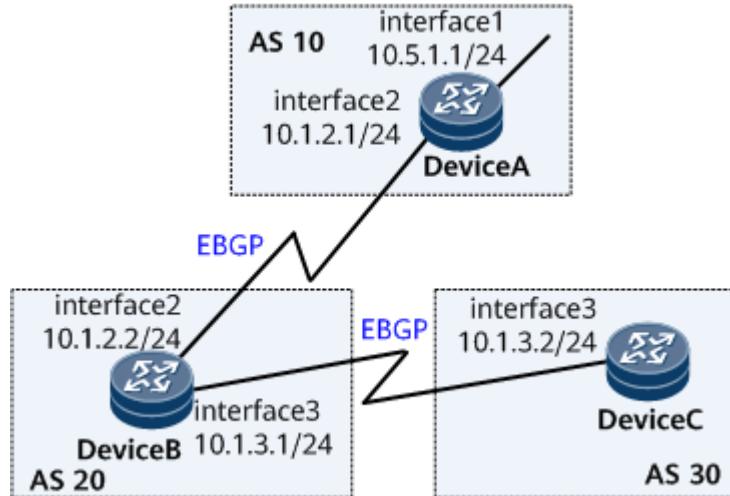
Enterprises A, B, and C belong to different ASs, and enterprise B's network communicates with the networks of the other two enterprises through EBGP connections. Enterprise A and C are rivals, and enterprise A requires that the routes it sends to enterprise B be transmitted only within enterprise B. In this situation, you can configure the community attribute on the router in enterprise A that sends routes to enterprise B.

In [Figure 1-431](#), EBGP connections are established between Device B and Device A, and between Device B and Device C. It is required that the routes advertised from AS 10 to AS 20 are not advertised to other ASs by AS 20. In this situation, configure the community attribute No\_Export on Device A.

**Figure 1-431** Configuring the BGP community attribute

 **NOTE**

Interfaces 1 through 3 in this example are GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Establish EBGP connections between Device A and Device B, and between Device B and Device C.
2. Configure a routing policy on Device A to advertise the community attribute No\_Export.

## Data Preparation

To complete the configuration, you need the following data:

- Router ID and AS number of Device A
- Router ID and AS number of Device B
- Router ID and AS number of Device C

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure EBGP connections.

# Configure Device A.

```
[~DeviceA] bgp 10
[*DeviceA-bgp] router-id 1.1.1.1
```

```
[*DeviceA-bgp] peer 10.1.2.2 as-number 20
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] network 10.5.1.0 255.255.255.0
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
```

# Configure Device B.

```
[~DeviceB] bgp 20
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.1.2.1 as-number 10
[*DeviceB-bgp] peer 10.1.3.2 as-number 30
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 30
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.1.3.1 as-number 20
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Check the routing table of Device B.

```
[~DeviceB] display bgp routing-table 10.5.1.0

BGP local router ID : 2.2.2.2
Local AS number : 20
Paths: 1 available, 1 best, 1 select
BGP routing table entry information of 10.5.1.0/24:
From: 10.1.2.1 (1.1.1.1)
Route Duration: 0d00h00m37s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 10, origin igp, MED 0, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
 10.1.2.1
 10.1.3.2
```

The command output shows that Device B advertises the received routes to Device C in AS 30.

# Check the routing table of Device C.

```
[~DeviceC] display bgp routing-table

BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network NextHop MED LocPrf PrefVal Path/Ogn
*> 10.5.1.0/24 10.1.3.1 0 20 10i
```

In the routing table, you can view that Device C learns the route to 10.5.1.0/24 from Device B.

### Step 3 Configure the BGP community attribute.

# Configure a routing policy on Device A to ensure that the routes advertised by Device A to Device B are not advertised by Device B to any other AS.

```
[~DeviceA] route-policy comm_policy permit node 10
```

```
[*DeviceA-route-policy] apply community no-export
[*DeviceA-route-policy] commit
[~DeviceA-route-policy] quit
```

# Apply the routing policy.

```
[~DeviceA] bgp 10
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 10.1.2.2 route-policy comm_policy export
[*DeviceA-bgp-af-ipv4] peer 10.1.2.2 advertise-community
[*DeviceA-bgp-af-ipv4] commit
```

# Check the routing table of Device B.

```
[~DeviceB] display bgp routing-table 10.5.1.0
```

```
BGP local router ID : 2.2.2.2
Local AS number : 20
Paths: 1 available, 1 best, 1 select
BGP routing table entry information of 10.5.1.0/24:
From: 10.1.2.1 (1.1.1.1)
Route Duration: 0d00h00m12s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.1
Qos information : 0x0
Community:no-export
AS-path 10, origin igp, MED 0, pref-val 0, valid, external, best, select, pre 255
Not advertised to any peers yet
```

The command output on DeviceB shows the configured community attribute and that the route to 10.5.1.0/24 has not been advertised to DeviceC.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.5.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0

bgp 10
router-id 1.1.1.1
peer 10.1.2.2 as-number 20

ipv4-family unicast
undo synchronization
network 10.5.1.0 255.255.255.0
peer 10.1.2.2 enable
peer 10.1.2.2 route-policy comm_policy export
peer 10.1.2.2 advertise-community

route-policy comm_policy permit node 10
apply community no-export

return
```

- Device B configuration file

```

sysname DeviceB
#
```

```
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0
#
bgp 20
router-id 2.2.2.2
peer 10.1.2.1 as-number 10
peer 10.1.3.2 as-number 30
#
ipv4-family unicast
undo synchronization
peer 10.1.2.1 enable
peer 10.1.3.2 enable
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
#
bgp 30
router-id 3.3.3.3
peer 10.1.3.1 as-number 20
#
ipv4-family unicast
undo synchronization
peer 10.1.3.1 enable
#
return
```

## Example for Configuring Prefix-based BGP ORF

Prefix-based BGP ORF is used to implement on-demand BGP route advertisement.

### Networking Requirements

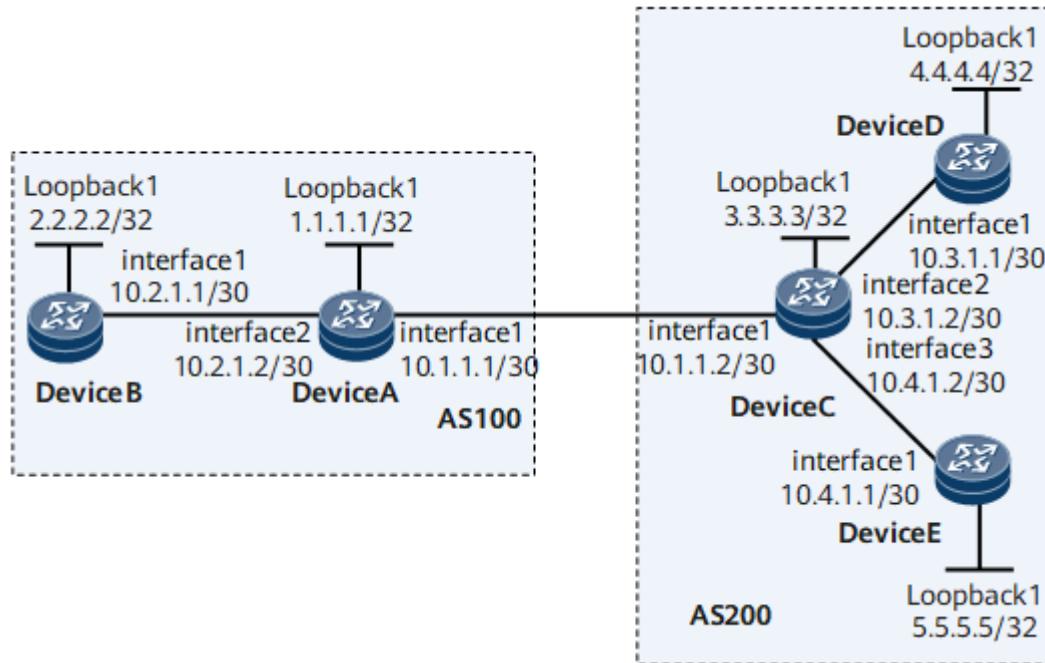
On the network shown in [Figure 1-432](#), Devices A and B are in AS 100. Devices C, D, and E are in AS 200. Device A requires Device C to send only routing information matching the import policy of Device A, but Device C does not want to maintain a separate export policy for Device A. Prefix-based BGP ORF can be configured in such a situation.

**Figure 1-432** Networking diagram for configuring prefix-based BGP ORF



NOTE

Interfaces 1 through 3 in this example are GE 1/0/0, GE 2/0/0, GE 3/0/0, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Establish an EBGP peer relationship between Devices A and C, and establish IBGP peer relationships between Devices A and B, between Devices C and D, and between Devices C and E.
2. Configure a prefix-based import policy on Device A, and enable prefix-based BGP ORF on Devices A and C.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs 1.1.1.1 and 2.2.2.2 and AS number 100 of Devices A and B respectively
- Router IDs 3.3.3.3, 4.4.4.4, and 5.5.5.5 and AS number 200 of Devices C, D, and E respectively

## Procedure

### Step 1 Configure an IP address for each interface.

Configure an IP address for each interface, as shown in [Figure 1-432](#). For details on configuration procedures, see corresponding configuration files.

**Step 2** Establish BGP peer relationships.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.2.1.1 as-number 100
[*DeviceA-bgp] peer 10.1.1.2 as-number 200
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] import-route direct
[*DeviceA-bgp-af-ipv4] quit
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.2.1.2 as-number 100
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.1.1.1 as-number 100
[*DeviceC-bgp] peer 10.3.1.1 as-number 200
[*DeviceC-bgp] peer 10.4.1.1 as-number 200
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] import-route direct
[*DeviceC-bgp-af-ipv4] quit
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 200
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 10.3.1.2 as-number 200
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

# Configure Device E.

```
[~DeviceE] bgp 200
[*DeviceE-bgp] router-id 5.5.5.5
[*DeviceE-bgp] peer 10.4.1.2 as-number 200
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

**Step 3** Configure a prefix-based import policy on Device A.

# Configure Device A.

```
[~DeviceA] ip ip-prefix 1 index 10 permit 10.3.1.0 24 less-equal 32
[*DeviceA] bgp 100
[*DeviceA-bgp] peer 10.1.1.2 ip-prefix 1 import
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# View routing information sent by Device C.

```
[~DeviceC] display bgp routing-table peer 10.1.1.1 advertised-routes

BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

| Total Number of Routes: 7 |          |     |        |         |          |
|---------------------------|----------|-----|--------|---------|----------|
| Network                   | NextHop  | MED | LocPrf | PrefVal | Path/Ogn |
| *> 3.3.3.3/32             | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.1.1.0/30            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.1.1.1/32            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.3.1.0/30            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.3.1.1/32            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.4.1.0/30            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.4.1.1/32            | 10.1.1.2 | 0   | 0      | 200?    |          |

# View routing information accepted by Device A.

```
[~DeviceA] display bgp routing-table peer 10.1.1.2 received-routes
```

```
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

| Total Number of Routes: 2 |          |     |        |         |          |
|---------------------------|----------|-----|--------|---------|----------|
| Network                   | NextHop  | MED | LocPrf | PrefVal | Path/Ogn |
| *> 10.3.1.0/30            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.3.1.1/32            | 10.1.1.2 | 0   | 0      | 200?    |          |

When prefix-based BGP ORF is not enabled, Device C sends seven routes, but Device A accepts only two routes because Device A applies the prefix-based import policy to the seven routes.

#### Step 4 Enable prefix-based BGP ORF.

# Enable prefix-based BGP ORF on DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 10.1.1.2 capability-advertise orf ip-prefix both
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Enable prefix-based BGP ORF on DeviceC.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] peer 10.1.1.1 capability-advertise orf ip-prefix both
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

#### Step 5 Verify the configuration.

# View prefix-based BGP ORF negotiation information on Device A.

```
[~DeviceA] display bgp peer 10.1.1.2 verbose
```

```
BGP Peer is 10.1.1.2, remote AS 200
Type: EBGP link
BGP version 4, Remote router ID 3.3.3.3
Update-group ID: 1
BGP current state: Established, Up for 00h00m01s
BGP current event: RecvRouteRefresh
BGP last state: OpenConfirm
BGP Peer Up count: 2
Received total routes: 0
Received active routes total: 0
Advertised total routes: 5
Port: Local - 179 Remote - 54545
Configured: Active Hold Time: 180 sec Keepalive Time:60 sec
```

```
Received : Active Hold Time: 180 sec
Negotiated: Active Hold Time: 180 sec Keepalive Time:60 sec
Peer optional capabilities:
Peer supports bgp multi-protocol extension
Peer supports bgp route refresh capability
Peer supports bgp outbound route filter capability
Support Address-Prefix: IPv4-UNC address-family, rfc-compatible, both
Peer supports bgp 4-byte-as capability
Address family IPv4 Unicast: advertised and received
Received: Total 3 messages
 Update messages 1
 Open messages 1
 KeepAlive messages 1
 Notification messages 0
 Refresh messages 1
Sent: Total 9 messages
 Update messages 5
 Open messages 2
 KeepAlive messages 1
 Notification messages 0
 Refresh messages 1
Authentication type configured: None
Last keepalive received: 2012-03-06 19:17:37 UTC-8:00
Last keepalive sent : 2012-03-06 19:17:37 UTC-8:00
Last update received: 2012-03-06 19:17:43 UTC-8:00
Last update sent : 2012-03-06 19:17:37 UTC-8:00
Minimum route advertisement interval is 30 seconds
Optional capabilities:
Route refresh capability has been enabled
Outbound route filter capability has been enabled
Enable Address-Prefix: IPv4-UNC address-family, rfc-compatible, both
4-byte-as capability has been enabled
Multi-hop ebgp has been enabled
Peer Preferred Value: 0
Routing policy configured:
No import update filter list
No export update filter list
Import prefix list is: 1
No export prefix list
No import route policy
No export route policy
No import distribute policy
No export distribute policy
```

# View routing information sent by Device C.

```
[~DeviceC] display bgp routing-table peer 10.1.1.1 advertised-routes
```

```
BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

| Total Number of Routes: 2 |          |     |        |         |          |
|---------------------------|----------|-----|--------|---------|----------|
| Network                   | NextHop  | MED | LocPrf | PrefVal | Path/Ogn |
| *> 10.3.1.0/30            | 10.1.1.2 | 0   | 0      | 200?    |          |
| *> 10.3.1.1/32            | 10.1.1.2 | 0   | 0      | 200?    |          |

# View routing information accepted by Device A.

```
[~DeviceA] display bgp routing-table peer 10.1.1.2 received-routes
```

```
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

| Total Number of Routes: 2 |             |          |     |        |         |          |
|---------------------------|-------------|----------|-----|--------|---------|----------|
|                           | Network     | NextHop  | MED | LocPrf | PrefVal | Path/Ogn |
| *>                        | 10.3.1.0/30 | 10.1.1.2 | 0   | 0      | 200?    |          |
| *>                        | 10.3.1.1/32 | 10.1.1.2 | 0   | 0      | 200?    |          |

After BGP ORF is enabled, Device C sends only two routes based on the prefix-based import policy provided by Device A, and Device A accepts only the two routes.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet2/0/0
ip address 10.2.1.2 255.255.255.252

interface GigabitEthernet1/0/0
ip address 10.1.1.1 255.255.255.252

interface LoopBack1
ip address 1.1.1.1 255.255.255.255

bgp 100
router-id 1.1.1.1
peer 10.1.1.2 as-number 200
peer 10.2.1.1 as-number 100

ipv4-family unicast
undo synchronization
import-route direct
peer 10.1.1.2 enable
peer 10.1.1.2 ip-prefix 1 import
peer 10.1.1.2 capability-advertise orf ip-prefix both
peer 10.2.1.1 enable

ip ip-prefix 1 index 10 permit 10.3.1.0 24 greater-equal 24 less-equal 32

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
ip address 10.2.1.1 255.255.255.252

interface LoopBack1
ip address 2.2.2.2 255.255.255.255

bgp 100
router-id 2.2.2.2
peer 10.2.1.2 as-number 100

ipv4-family unicast
undo synchronization
peer 10.2.1.2 enable

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet2/0/0
ip address 10.3.1.2 255.255.255.252

interface GigabitEthernet1/0/0
ip address 10.1.1.2 255.255.255.252

interface GigabitEthernet3/0/0
ip address 10.4.1.2 255.255.255.252

interface LoopBack1
ip address 3.3.3.3 255.255.255.255

bgp 200
router-id 3.3.3.3
peer 10.1.1.1 as-number 100
peer 10.3.1.1 as-number 200
peer 10.4.1.1 as-number 200

ipv4-family unicast
undo
synchronization
import-route direct
peer 10.1.1.1 enable
peer 10.1.1.1 capability-advertise orf ip-prefix both
peer 10.3.1.1 enable
peer 10.4.1.1 enable

return
```

- DeviceD configuration file

```

sysname DeviceD

interface GigabitEthernet1/0/0
ip address 10.3.1.1 255.255.255.252

interface LoopBack1
ip address 4.4.4.4 255.255.255.255

bgp 200
router-id 4.4.4.4
peer 10.3.1.2 as-number 200

ipv4-family unicast
undo
synchronization
peer 10.3.1.2 enable

return
```

- DeviceE configuration file

```

sysname DeviceE

interface GigabitEthernet1/0/1
ip address 10.4.1.1 255.255.255.252

interface LoopBack1
ip address 5.5.5.5 255.255.255.255

bgp 200
router-id 5.5.5.5
peer 10.4.1.2 as-number 200

ipv4-family unicast
undo synchronization
peer 10.4.1.2 enable
```

```

return
```

## Example for Configuring BGP Route Dampening

Configuring BGP route dampening can improve network stability.

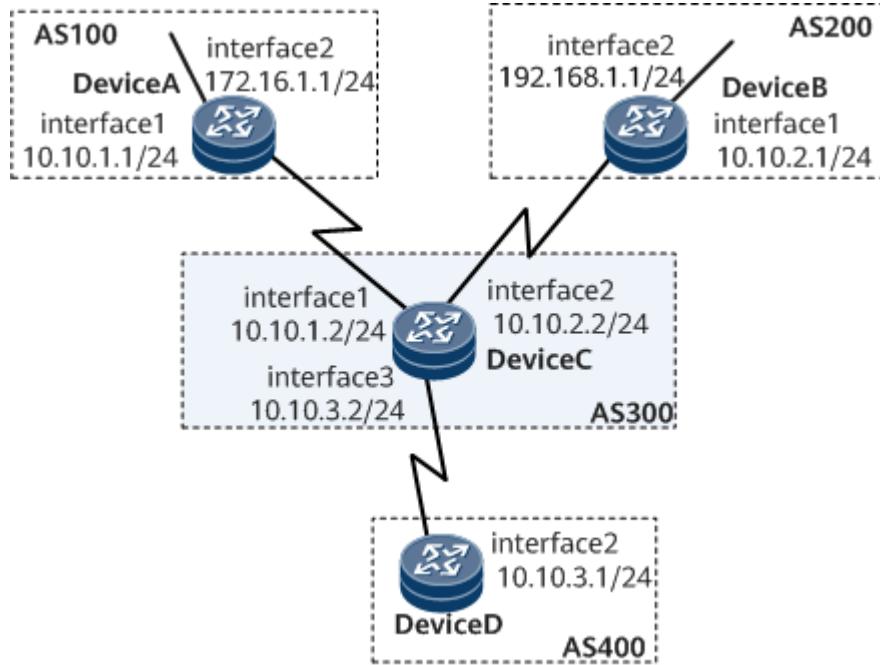
## Networking Requirements

In [Figure 1-433](#), all routers are configured with BGP; DeviceA resides in AS 100; DeviceB resides in AS 200; DeviceC resides in AS 300; DeviceD resides in AS 400. EBGP runs between Device C and Device A, between Device C and Device B, and between Device C and Device D. For routes from different EBGP peers, DeviceC applies different route dampening policies. BGP route dampening suppresses unstable routes and improves network stability.

**Figure 1-433** Configuring BGP route dampening



In this example, interface1, interface2, and interface3 represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

During the configuration, note the following:

- BGP route dampening takes effect only on EBGP routes.
- Set a small **MaxSuppressTime** value for routes with smaller destination address masks.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Establish EBGP connections between DeviceA and DeviceC, between DeviceB and DeviceC, and between DeviceD and DeviceC.
2. Configure route dampening policies on Device C and then check routes.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A, Device B, Device C, and Device D
- Name of the route flap dampening policy to be applied to Device C

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

**Step 2** Configure BGP connections.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.10.1.2 as-number 300
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] network 172.16.1.0 24
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.10.2.2 as-number 300
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] network 192.168.1.0 24
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 300
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.10.1.1 as-number 100
[*DeviceC-bgp] peer 10.10.2.1 as-number 200
[*DeviceC-bgp] peer 10.10.3.1 as-number 400
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 400
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 10.10.3.2 as-number 300
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

# Check the BGP peers of Device C.

```
[*DeviceC] display bgp peer

BGP local router ID : 3.3.3.3
Local AS number : 300
Total number of peers : 3 Peers in established state : 3

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
10.10.1.1 4 100 3 3 0 00:00:01 Established 0
10.10.2.1 4 200 3 3 0 00:00:00 Established 0
10.10.3.1 4 400 3 3 0 00:00:01 Established 0
```

The command output on DeviceC shows that the BGP connection status of with each peer is **Established**.

**Step 3** Configure BGP route dampening policies.

```
Configure an IP prefix list named prefix-a on DeviceC to permit the routes with prefix 172.16.1.0/24.
```

```
[~DeviceC] ip ip-prefix prefix-a index 10 permit 172.16.1.0 24
[*DeviceC] commit
```

```
Configure an IP prefix list named prefix-b on DeviceC to permit the routes with prefix 192.168.1.0/24.
```

```
[~DeviceC] ip ip-prefix prefix-b index 20 permit 192.168.1.0 24
[*DeviceC] commit
```

```
Configure a route-policy named dampen-policy on Device C and then apply different route dampening policies to the routes with different prefix lengths.
```

```
[~DeviceC] route-policy dampen-policy permit node 10
[*DeviceC-route-policy] if-match ip-prefix prefix-a
[*DeviceC-route-policy] apply dampening 10 1000 2000 5000
[*DeviceC-route-policy] commit
[~DeviceC-route-policy] quit
[~DeviceC] route-policy dampen-policy permit node 20
[*DeviceC-route-policy] if-match ip-prefix prefix-b
[*DeviceC-route-policy] apply dampening 10 800 3000 10000
[*DeviceC-route-policy] commit
[~DeviceC-route-policy] quit
```

```
Apply route dampening policies to flapping routes.
```

```
[*DeviceC] bgp 300
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] dampening route-policy dampen-policy
[*DeviceC-bgp-af-ipv4] commit
[~DeviceC-bgp] quit
```

```
Check the configured route dampening parameters on DeviceC.
```

```
[~DeviceC] display bgp routing-table dampening parameter
```

```
Maximum Suppress Time(in second) : 3973
Ceiling Value : 16000
Reuse Value : 750
HalfLife Time(in second) : 900
Suppress-Limit : 2000
Route-policy : dampen-policy
```

----End

## Configuration Files

- Device A configuration file

```
#
```

```
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.10.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.1.1 255.0.0.0
#
bgp 100
router-id 1.1.1.1
peer 10.10.1.2 as-number 300
#
ipv4-family unicast
undo synchronization
network 172.16.1.0 255.255.255.0
peer 10.10.1.2 enable
#
return
```

- **Device B configuration file**

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.10.2.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
bgp 200
router-id 2.2.2.2
peer 10.10.2.2 as-number 300
#
ipv4-family unicast
undo synchronization
network 192.168.1.0 255.255.255.0
peer 10.10.2.2 enable
#
return
```

- **Device C configuration file**

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.10.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.10.2.2 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.10.3.2 255.255.255.0
#
bgp 300
router-id 3.3.3.3
peer 10.10.1.1 as-number 100
peer 10.10.2.1 as-number 200
peer 10.10.3.1 as-number 400
#
ipv4-family unicast
undo synchronization
dampening route-policy dampen-policy
peer 10.10.1.1 enable
```

```
peer 10.10.2.1 enable
peer 10.10.3.1 enable
#
ip ip-prefix prefix-a index 10 permit 172.16.1.0 24
#
ip ip-prefix prefix-b index 20 permit 192.168.1.0 24
#
route-policy dampen-policy permit node 10
if-match ip-prefix prefix-a
apply dampening 10 1000 2000 5000
#
route-policy dampen-policy permit node 20
if-match ip-prefix prefix-b
apply dampening 10 800 3000 10000
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.10.3.1 255.255.255.0
#
bgp 400
router-id 4.4.4.4
peer 10.10.3.2 as-number 300
#
ipv4-family unicast
undo synchronization
peer 10.10.3.2 enable
#
return
```

## Example for Configuring BGP Default Route Advertisement

By controlling the advertising of default routes, you can specify traffic from a specific path to enter ASs.

## Networking Requirements

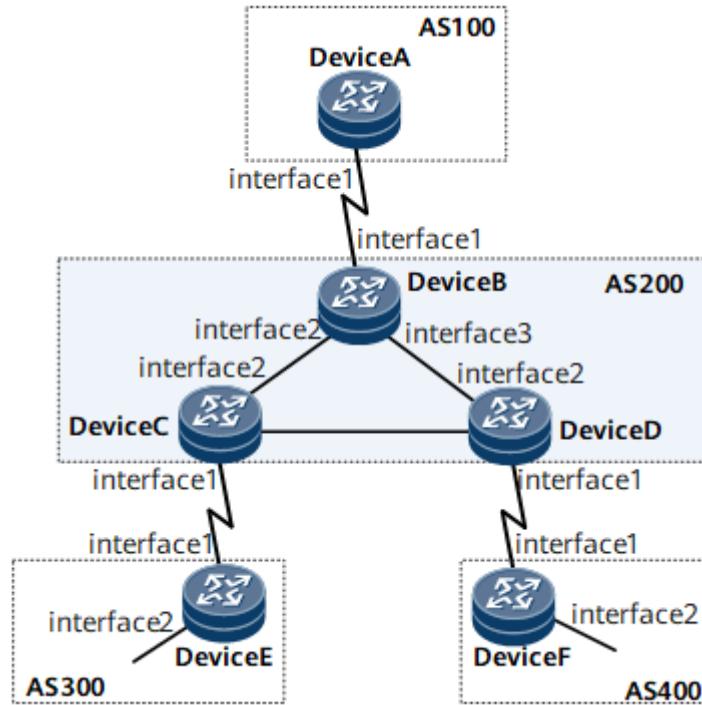
In [Figure 1-434](#), all routers run BGP. To ensure that the traffic that leaves AS 200 is forwarded by Device E and Device F, EBGP connections are established between Device A and Device B, between Device C and Device E, and between Device D and Device F; IBGP connections are established between Device B and Device C, and between Device B and Device D.

**Figure 1-434** Configuring BGP default route advertisement



### NOTE

Interfaces 1 through 3 in this example are GE 1/0/0, GE 2/0/0, GE 3/0/0, respectively.



| Device Name | Interface  | IP Address   |
|-------------|------------|--------------|
| Device A    | GE 1/0/0   | 10.20.1.1/24 |
|             | Loopback 0 | 1.1.1.1/32   |
| Device B    | GE 1/0/0   | 10.20.1.2/24 |
|             | GE 2/0/0   | 10.0.1.1/24  |
|             | GE 3/0/0   | 10.0.3.2/24  |
|             | Loopback 0 | 2.2.2.2/32   |
| Device C    | GE 1/0/0   | 10.20.2.2/24 |
|             | GE 2/0/0   | 10.0.1.2/24  |
|             | GE 3/0/0   | 10.0.2.1/24  |
|             | Loopback 0 | 3.3.3.3/32   |
| Device D    | GE 1/0/0   | 10.20.3.2/24 |
|             | GE 2/0/0   | 10.0.3.1/24  |
|             | GE 3/0/0   | 10.0.2.2/24  |
|             | Loopback 0 | 4.4.4.4/32   |
| Device E    | GE 1/0/0   | 10.20.2.1/24 |
|             | GE 2/0/0   | 10.1.1.1/24  |
|             | Loopback 0 | 5.5.5.5/32   |

| Device Name | Interface  | IP Address   |
|-------------|------------|--------------|
| Device F    | GE 1/0/0   | 10.20.3.1/24 |
|             | GE 2/0/0   | 10.2.1.1/24  |
|             | Loopback 0 | 6.6.6.6/32   |

## Precautions

During the configuration, note the following:

- A default route can represent all routes on the entire network. For example, in a stub AS scenario, the local device can use a default route, instead of advertising network-wide routes, to forward traffic to an external network. A default route can also represent all routes except specific ones.
- During the establishment of a peer relationship, if the IP address of the specified peer is a loopback interface address or a sub-interface address, the **peer connect-interface** command must be run on both ends to ensure correct connection.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure OSPF on Device B, Device C, and Device D.
2. Establish EBGP connections between Device A and Device B, between Device C and Device E, and between Device D and Device F.
3. Establish IBGP connections between Device B and Device C, and between Device B and Device D.
4. Configure an import routing policy on Device C to accept only default routes.
5. Configure an import routing policy on Device D to accept default routes and all specific routes, and then set Local\_Pref values for the accepted default routes.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A, Device B, Device C, Device D, Device E, and Device F
- Names of the import routing policies to be configured on Device C and Device D
- Local\_Pref values to be set for the accepted default routes on Device D

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

**Step 2** Configure OSPF.

# Configure Device B.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0] network 10.0.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0] network 10.0.3.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0] network 2.2.2.2 0.0.0.0
[*DeviceB-ospf-1-area-0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0] quit
[~DeviceB-ospf-1] quit
```

# Configure Device C.

```
[~DeviceC] ospf 1
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0] network 10.0.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0] network 10.0.2.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0] network 3.3.3.3 0.0.0.0
[*DeviceC-ospf-1-area-0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0] quit
[~DeviceC-ospf-1] quit
```

# Configure Device D.

```
[~DeviceD] ospf 1
[*DeviceD-ospf-1] area 0
[*DeviceD-ospf-1-area-0.0.0] network 10.0.2.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0] network 10.0.3.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0] network 4.4.4.4 0.0.0.0
[*DeviceD-ospf-1-area-0.0.0] commit
[~DeviceD-ospf-1-area-0.0.0] quit
[~DeviceD-ospf-1] quit
```

**Step 3** Configure BGP connections.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.20.1.2 as-number 200
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.20.1.1 as-number 100
[*DeviceB-bgp] network 10.20.1.0 24
[*DeviceB-bgp] peer 3.3.3.3 as-number 200
[*DeviceB-bgp] peer 3.3.3.3 connect-interface LoopBack0
[*DeviceB-bgp] peer 4.4.4.4 as-number 200
[*DeviceB-bgp] peer 4.4.4.4 connect-interface LoopBack0
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.20.2.1 as-number 300
[*DeviceC-bgp] network 10.20.2.0 24
```

```
[*DeviceC-bgp] peer 2.2.2.2 as-number 200
[*DeviceC-bgp] peer 2.2.2.2 connect-interface LoopBack0
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 200
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 10.20.3.1 as-number 400
[*DeviceD-bgp] network 10.20.3.0 24
[*DeviceD-bgp] peer 2.2.2.2 as-number 200
[*DeviceD-bgp] peer 2.2.2.2 connect-interface LoopBack0
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

# Configure Device E.

```
[~DeviceE] bgp 300
[*DeviceE-bgp] router-id 5.5.5.5
[*DeviceE-bgp] peer 10.20.2.2 as-number 200
[*DeviceE-bgp] network 10.1.1.0 24
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

# Configure Device F.

```
[~DeviceF] bgp 400
[*DeviceF-bgp] router-id 6.6.6.6
[*DeviceF-bgp] peer 10.20.3.2 as-number 200
[*DeviceF-bgp] network 10.2.1.0 24
[*DeviceF-bgp] commit
[~DeviceF-bgp] quit
```

#### Step 4 Configure Device E and Device F to advertise default routes.

# Configure Device E to advertise default routes.

```
[~DeviceE-bgp] ipv4-family unicast
[*DeviceE-bgp-af-ipv4] peer 10.20.2.2 default-route-advertise
[*DeviceE-bgp-af-ipv4] commit
```

# Configure Device F to advertise default routes.

```
[~DeviceF-bgp] ipv4-family unicast
[*DeviceF-bgp-af-ipv4] peer 10.20.3.2 default-route-advertise
[*DeviceF-bgp-af-ipv4] commit
```

# Check the routing table of Device B.

```
[~DeviceB] display bgp routing-table

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 7
 Network NextHop MED LocPrf PrefVal Path/Ogn
*>i 0.0.0.0 10.20.2.1 0 100 0 300i
* i 10.20.3.1 0 100 0 400i
*>i 10.1.1.0/24 10.20.2.1 0 100 0 300i
*>i 10.2.1.0/24 10.20.3.1 0 100 0 400i
*> 10.20.1.0 0.0.0.0 0 0 i
*>i 10.20.2.0 3.3.3.3 0 100 0 i
*>i 10.20.3.0 4.4.4.4 0 100 0 i
```

The command output shows that Device B has received the default routes and all specific routes of AS 300 and AS 400.

**Step 5** Configure import routing policies.

# Configure an IP prefix list named **default** on Device C to accept only default routes.

```
[~DeviceC] ip ip-prefix default index 10 permit 0.0.0.0 0
[*DeviceC] commit
[*DeviceC] bgp 200
[*DeviceC-bgp] peer 10.20.2.1 ip-prefix default import
[*DeviceC-bgp] commit
```

# Configure a route-policy named **set-default-low** on Device D to accept default routes and all specific routes, and set Local\_Pref values for the accepted default routes.

```
[*DeviceD] ip ip-prefix default index 10 permit 0.0.0.0 0
[~DeviceD] ip as-path-filter 10 permit ^(400_)+$
[*DeviceD] ip as-path-filter 10 permit ^(400_)+_[0-9]+$
[*DeviceD] route-policy set-default-low permit node 10
[*DeviceD-route-policy] if-match ip-prefix default
[*DeviceD-route-policy] apply local-preference 80
[*DeviceD-route-policy] quit
[*DeviceD] route-policy set-default-low permit node 20
[*DeviceD-route-policy] quit
[*DeviceD] commit
[~DeviceD] bgp 200
[*DeviceD-bgp] peer 10.20.3.1 as-path-filter 10 import
[*DeviceD-bgp] peer 10.20.3.1 route-policy set-default-low import
[*DeviceD-bgp] commit
```

# Check the routing table of Device B.

```
[~DeviceB] display bgp routing-table

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
 Network NextHop MED LocPrf PrefVal Path/Ogn
*>i 0.0.0.0 10.20.2.1 0 100 0 300i
* i 10.20.3.1 0 80 0 400i
*>i 10.2.1.0/24 10.20.3.1 0 100 0 400i
*> 10.20.1.0 0.0.0.0 0 0 i
*>i 10.20.2.0 3.3.3.3 0 100 0 i
*>i 10.20.3.0 4.4.4.4 0 100 0 i
```

The command output shows that Device B has received only the default routes of AS 300 and the default routes and all specific routes of AS 400 and that the Local\_Pref of the accepted default routes destined of AS 400 has been set to 80.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.1.1 255.255.255.0
#
interface LoopBack0
ip address 1.1.1.1 255.255.255.255
#
bgp 100
router-id 1.1.1.1
peer 10.20.1.2 as-number 200
#
ipv4-family unicast
peer 10.20.1.2 enable
#
return
```

- Device B configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.1.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.0.3.2 255.255.255.0
#
interface LoopBack0
ip address 2.2.2.2 255.255.255.255
#
bgp 200
router-id 2.2.2.2
peer 3.3.3.3 as-number 200
peer 3.3.3.3 connect-interface LoopBack0
peer 4.4.4.4 as-number 200
peer 4.4.4.4 connect-interface LoopBack0
peer 10.20.1.1 as-number 100
#
ipv4-family unicast
network 10.20.1.0 255.255.255.0
peer 3.3.3.3 enable
peer 4.4.4.4 enable
peer 10.20.1.1 enable
#
ospf 1
area 0.0.0
network 2.2.2.2 0.0.0.0
network 10.0.1.0 0.0.0.255
network 10.0.3.0 0.0.0.255
#
return
```

- Device C configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.2.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.1.2 255.255.255.0
#
interface GigabitEthernet3/0/0
```

```
undo shutdown
ip address 10.0.2.1 255.255.255.0
#
interface LoopBack0
ip address 3.3.3.3 255.255.255.255
#
bgp 200
router-id 3.3.3.3
peer 2.2.2.2 as-number 200
peer 2.2.2.2 connect-interface LoopBack0
peer 10.20.2.1 as-number 300
#
ipv4-family unicast
network 10.20.2.0 255.255.255.0
peer 2.2.2.2 enable
peer 10.20.2.1 enable
peer 10.20.2.1 ip-prefix default import
#
ospf 1
area 0.0.0.0
network 3.3.3.3 0.0.0.0
network 10.0.1.0 0.0.0.255
network 10.0.2.0 0.0.0.255
#
ip ip-prefix default index 10 permit 0.0.0.0 0
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.3.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.3.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 10.0.2.2 255.255.255.0
#
interface LoopBack0
ip address 4.4.4.4 255.255.255.255
#
bgp 200
router-id 4.4.4.4
peer 2.2.2.2 as-number 200
peer 2.2.2.2 connect-interface LoopBack0
peer 10.20.3.1 as-number 400
#
ipv4-family unicast
network 10.20.3.0 255.255.255.0
peer 2.2.2.2 enable
peer 10.20.3.1 enable
peer 10.20.3.1 as-path-filter 10 import
peer 10.20.3.1 route-policy set-default-low import
#
ospf 1
area 0.0.0.0
network 4.4.4.4 0.0.0.0
network 10.0.2.0 0.0.0.255
network 10.0.3.0 0.0.0.255
#
ip ip-prefix default index 10 permit 0.0.0.0 0
#
ip as-path-filter 10 permit ^(400_)+$
ip as-path-filter 10 permit ^^(400_)+_[0-9]+$
```

```

route-policy set-default-low permit node 10
if-match ip-prefix default
apply local-preference 80

route-policy set-default-low permit node 20

return
```

- Device E configuration file

```

sysname DeviceE

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.2.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

interface LoopBack0
ip address 5.5.5.5 255.255.255.255

bgp 300
router-id 5.5.5.5
peer 10.20.2.2 as-number 200

ipv4-family unicast
network 10.1.1.0 255.255.255.0
peer 10.20.2.2 enable
peer 10.20.2.2 default-route-advertise

return
```

- Device F configuration file

```

sysname DeviceF

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.3.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0

interface LoopBack0
ip address 6.6.6.6 255.255.255.255

bgp 400
router-id 6.6.6.6
peer 10.20.3.2 as-number 200

ipv4-family unicast
network 10.2.1.0 255.255.255.0
peer 10.20.3.2 enable
peer 10.20.3.2 default-route-advertise

return
```

## Example for Configuring BGP Load Balancing

Configuring load balancing can fully utilize network resources and reduce network congestion.

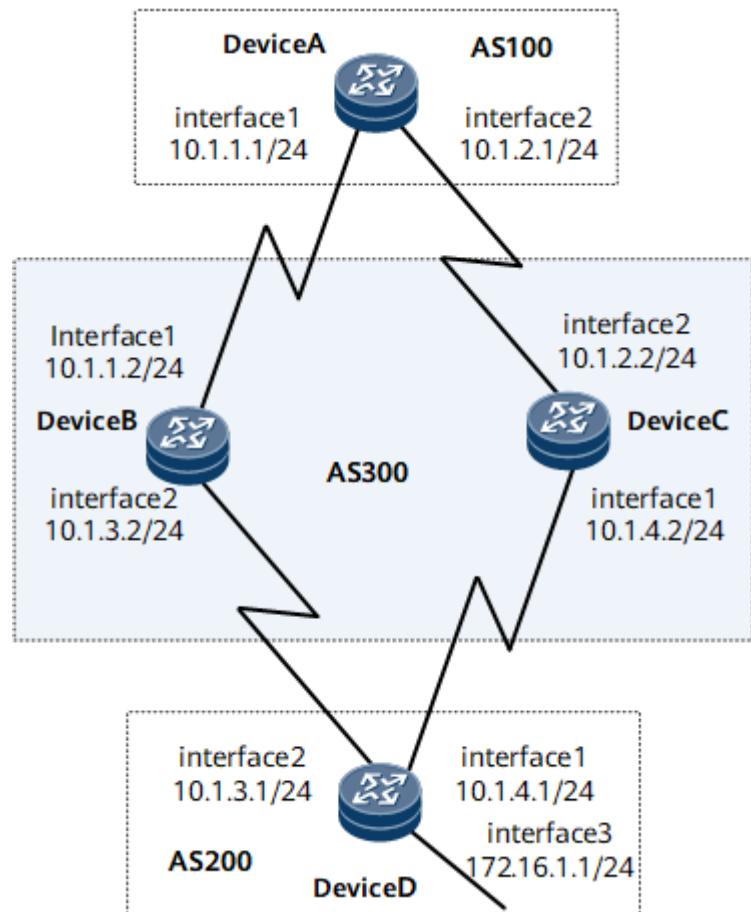
## Networking Requirements

In [Figure 1-435](#), all routers run BGP; Device A resides in AS 100; Device B and Device C reside in AS 300; Device D resides in AS 200. EBGP connections are established between Device A and Device B, between Device A and Device C, between Device D and Device B, and between Device D and Device C. On Device A, there are two BGP routes to 172.16.1.0/24. Traffic to 172.16.1.0/24 can reach the destination through Device B and Device C. It is required that BGP load balancing be configured to fully utilize network resources and reduce network congestion.

**Figure 1-435** Configuring BGP load balancing

 NOTE

Interfaces 1 through 2 in this example are GE 1/0/0 and GE 2/0/0, respectively.



## Precautions

During the configuration process, note the following:

- Route load balancing can be implemented by configuring BGP attributes, for example, configuring the device to ignore the comparison of IGP metrics. Ensure that no routing loops occur when configuring BGP attributes to implement load balancing.

- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Establish EBGP connections between Device A and Device B, and between Device A and Device C, and establish an IBGP connection between Device B and Device C.
2. Establish EBGP connections between Device D and Device B, and between Device D and Device C.
3. Configure load balancing on Device A and then check routes.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A, Device B, Device C, and Device D
- Number of routes for load balancing

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

**Step 2** Configure BGP connections.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.1.1.2 as-number 300
[*DeviceA-bgp] peer 10.1.2.2 as-number 300
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 300
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.1.1.1 as-number 100
[*DeviceB-bgp] peer 10.1.3.1 as-number 200
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 300
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.1.2.1 as-number 100
[*DeviceC-bgp] peer 10.1.4.1 as-number 200
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 200
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 10.1.3.2 as-number 300
```

```
[*DeviceD-bgp] peer 10.1.4.2 as-number 300
[*DeviceD-bgp] ipv4-family unicast
[*DeviceD-bgp-af-ipv4] network 172.16.1.0 255.255.255.0
[*DeviceD-bgp-af-ipv4] commit
[~DeviceD-bgp-af-ipv4] quit
[~DeviceD-bgp] quit
```

# Check the routing table of Device A.

```
[~DeviceA] display bgp routing-table 172.16.1.0 24

BGP local router ID : 1.1.1.1
Local AS number : 100
Paths : 2 available, 1 best, 1 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.1.2 (2.2.2.2)
Route Duration: 0d00h00m50s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 200 300, origin igp, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
 10.1.1.2
 10.1.2.2

BGP routing table entry information of 172.16.1.0/24:
From: 10.1.2.2 (3.3.3.3)
Route Duration: 0d00h00m51s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 200 300, origin igp, pref-val 0, valid, external, pre 255, not preferred for router ID
Not advertised to any peers yet
```

The command output shows that there are two valid routes from Device A to 172.16.1.0/24. The route with the next hop 10.1.1.2 is the optimal route because the router ID of Device B is smaller.

**Step 3** Configure load balancing.

# Configure load balancing on Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] maximum load-balancing 2
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

**Step 4** Verify the configuration.

# Check the routing table of Device A.

```
[~DeviceA] display bgp routing-table 172.16.1.0 24

BGP local router ID : 1.1.1.1
Local AS number : 100
Paths : 2 available, 1 best, 2 select
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.1.2 (2.2.2.2)
Route Duration: 0d00h03m55s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.1.1.2
Qos information : 0x0
AS-path 200 300, origin igp, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
 10.1.1.2
 10.1.2.2
```

```
BGP routing table entry information of 172.16.1.0/24:
From: 10.1.2.2 (3.3.3.3)
Route Duration: 0d00h03m56s
Direct Out-interface: GigabitEthernet2/0/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path 200 300, origin igp, pref-val 0, valid, external, select, pre 255, not preferred for router ID
Not advertised to any peers yet
```

The routing table shows that the BGP route 172.16.1.0/24 has two next hops: 10.1.1.2 and 10.1.2.2.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0

bgp 100
router-id 1.1.1.1
peer 10.1.1.2 as-number 300
peer 10.1.2.2 as-number 300

ipv4-family unicast
maximum load-balancing 2
peer 10.1.1.2 enable
peer 10.1.2.2 enable

return
```

- Device B configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0

bgp 300
router-id 2.2.2.2
peer 10.1.1.1 as-number 100
peer 10.1.3.1 as-number 200

ipv4-family unicast
undo synchronization
peer 10.1.1.1 enable
peer 10.1.3.1 enable

return
```

- Device C configuration file

```

sysname DeviceC
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
bgp 300
router-id 3.3.3.3
peer 10.1.2.1 as-number 100
peer 10.1.4.1 as-number 200
#
ipv4-family unicast
undo synchronization
peer 10.1.2.1 enable
peer 10.1.4.1 enable
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
bgp 200
router-id 4.4.4.4
peer 10.1.3.2 as-number 300
peer 10.1.4.2 as-number 300
#
ipv4-family unicast
undo synchronization
network 172.16.1.0 255.255.255.0
peer 10.1.3.2 enable
peer 10.1.4.2 enable
#
return
```

## Example for Configuring BGP Next Hop Recursion Based on a Routing Policy

Configuring BGP next hop recursion based on a routing policy prevents traffic loss in case of route changes.

## Networking Requirements

As shown in [Figure 1-436](#), OSPF is used as the IGP in AS 100. An IBGP peer relationship is established between loopback0 interfaces of DeviceA and DeviceB, and between loopback0 interfaces of DeviceA and DeviceC. DeviceB and DeviceC advertise the BGP route 10.20.1.0/24 to DeviceA. Because the router ID of Device B is smaller, Device A chooses the route 10.20.1.0/24 that is learned from Device B as the optimal route, with the original next hop of 2.2.2.2/32.

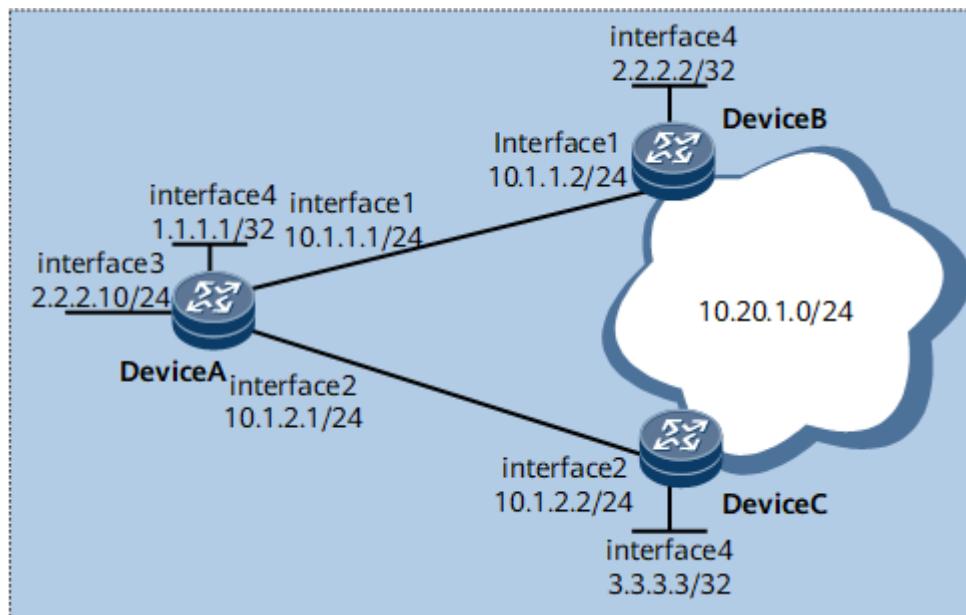
In most cases, Device A recurses the next hop of the BGP route destined for 10.20.1.0/24 to an IGP route destined for 2.2.2.2/32 with GE1/0/0 as the outbound

interface. When Device B is faulty, Device A deletes the IGP route destined for 2.2.2.2/32 immediately. However, Device A still considers the BGP route with 2.2.2.2/32 being the original next hop as the optimal route because it does not know the BGP route change before the BGP hold timer expires. Based on the longest matching rule, Device A mistakenly recurses the BGP route destined for 10.20.1.0/24 to the direct route destined for 2.2.2.10/24, with GE1/0/2 as the outbound interface, causing traffic loss.

**Figure 1-436** Networking diagram for configuring BGP next hop recursion based on a routing policy

 NOTE

Interfaces 1 through 4 in this example are GE 1/0/0, GE 1/0/1, GE 1/0/2, Loopback0, respectively.



To prevent traffic loss, configure BGP next hop recursion based on a routing policy on Device A to control the recursive routes. In this example, only the recursive routes with a mask length of 32 bits match the routing policy, and those that do not match the routing policy are considered unreachable. As such, when DeviceB is faulty, DeviceA can rapidly detect the route change and re-select a correct BGP route with the original next hop of 3.3.3.3/32, preventing traffic loss.

## Precautions

When configuring BGP next hop recursion based on a routing policy, note the following:

- Ensure that all desirable recursive routes match the routing policy. Otherwise, BGP routes may be considered unreachable, unable to guide traffic forwarding.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure OSPF on Device A, Device B, and Device C to enable the devices in AS 100 to communicate with each other.
2. Establish an IBGP peer relationship between Loopback 0s of Device A and Device B, and between Loopback 0s of Device A and Device C.
3. Enable Device B and Device C to advertise a BGP route destined for 10.20.1.0/24 to Device A.
4. Configure BGP next hop recursion based on a routing policy on Device A. This configuration allows Device A to know the route change in time when Device B is faulty and re-select a correct BGP route, preventing traffic loss.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs of Device A, Device B, and Device C (1.1.1.1, 2.2.2.2, and 3.3.3.3, respectively) and AS number (100)
- Routing policy (**np-by-rp**) configured on Device A to control route recursion.

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see [Configuration File](#).

**Step 2** Configure OSPF in AS 100.

# Configure Device A.

```
[~DeviceA] ospf 1
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0] network 1.1.1.1 0.0.0.0
[*DeviceA-ospf-1-area-0.0.0] network 10.1.0.0 0.0.255.255
[*DeviceA-ospf-1-area-0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0] quit
[~DeviceA-ospf-1]
```

# Configure Device B.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0] network 2.2.2.2 0.0.0.0
[*DeviceB-ospf-1-area-0.0.0] network 10.1.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0] quit
[~DeviceB-ospf-1]
```

# Configure Device C.

```
[~DeviceC] ospf 1
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0] network 3.3.3.3 0.0.0.0
[*DeviceC-ospf-1-area-0.0.0] network 10.1.2.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0] quit
[~DeviceC-ospf-1]
```

**Step 3** Establish IBGP connections.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 2.2.2.2 as-number 100
[*DeviceA-bgp] peer 3.3.3.3 as-number 100
[*DeviceA-bgp] peer 2.2.2.2 connect-interface LoopBack 0
[*DeviceA-bgp] peer 3.3.3.3 connect-interface LoopBack 0
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 1.1.1.1 as-number 100
[*DeviceB-bgp] peer 1.1.1.1 connect-interface LoopBack 0
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 1.1.1.1 as-number 100
[*DeviceC-bgp] peer 1.1.1.1 connect-interface LoopBack 0
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

**Step 4** Enable Device B and Device C to advertise a BGP route destined for 10.20.1.0/24.

# Configure Device B.

```
[~DeviceB] ip route-static 10.20.1.0 24 NULL 0
[*DeviceB] commit
[~DeviceB] bgp 100
[*DeviceB-bgp] import-route static
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] ip route-static 10.20.1.0 24 NULL 0
[*DeviceC] commit
[~DeviceC] bgp 100
[*DeviceC-bgp] import-route static
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

**Step 5** Configure BGP next hop recursion based on a routing policy on Device A.

# Configure Device A.

```
[~DeviceA] ip ip-prefix np-by-rp-ip index 10 permit 0.0.0.0 32
[*DeviceA] route-policy np-by-rp permit node 10
[*DeviceA-route-policy] if-match ip-prefix np-by-rp-ip
[*DeviceA-route-policy] quit
[*DeviceA] bgp 100
[*DeviceA-bgp] nexthop recursive-lookup route-policy np-by-rp
[*DeviceA-bgp] quit
[*DeviceA] commit
```

**Step 6** Verify the configuration.

# Display detailed information about the BGP route destined for 10.20.1.0/24 on Device A when Device B is running properly.

```
[~DeviceA] display bgp routing-table 10.20.1.0 24
```

```
BGP local router ID : 1.1.1.1
```

```
Local AS number : 100
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.20.1.0/24:
From: 2.2.2.2 (2.2.2.2) Route Duration: 0d00h00m36s
Relay IP Nexthop: 10.1.1.2
Relay IP Out-interface: GigabitEthernet1/0/0
Original nexthop: 2.2.2.2
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Not advertised to any peer yet

BGP routing table entry information of 10.20.1.0/24:
From: 3.3.3.3 (3.3.3.3) Route Duration: 0d02h53m45s
Relay IP Nexthop: 10.1.2.2
Relay IP Out-interface: GigabitEthernet1/0/1
Original nexthop: 3.3.3.3
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, pre 255,
not preferred for router ID
Not advertised to any peers yet
```

# Run the **shutdown** command on GE 1/0/0 of Device B to simulate a fault on Device B.

```
[~DeviceB] interface GigabitEthernet 1/0/0
[~DeviceB-GigabitEthernet1/0/0] shutdown
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
```

# Display detailed information about the BGP route destined for 10.20.1.0/24 on Device A.

```
[~DeviceA] display bgp routing-table 10.20.1.0 24

BGP local router ID : 1.1.1.1
Local AS number : 100
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 10.20.1.0/24:
From: 3.3.3.3 (3.3.3.3) Route Duration: 0d03h10m58s
Relay IP Nexthop: 10.1.2.2
Relay IP Out-interface: GigabitEthernet1/0/1
Original nexthop: 3.3.3.3
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Not advertised to any peer yet

BGP routing table entry information of 10.20.1.0/24:
From: 2.2.2.2 (2.2.2.2) Route Duration: 0d00h00m50s
Relay IP Nexthop: 0.0.0.0
Relay IP Out-interface:
Original nexthop: 2.2.2.2
Qos information : 0x0
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, internal, pre 255
Not advertised to any peers yet
```

After DeviceB becomes faulty, the route which is destined for 10.20.1.0/24 and has the original next hop of 2.2.2.2/32 recurses to the direct route destined for 2.2.2.10/24. However, because the direct route destined for 2.2.2.10/24 is not a specific route with a 32-bit mask, this direct route does not match the route-policy **np-by-rp**. As a result, the recursive route is considered unreachable. Then, the correct route with 3.3.3.3/32 as the original next hop is selected by BGP.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.2.1 255.255.255.0

interface GigabitEthernet1/0/2
undo shutdown
ip address 2.2.2.10 255.255.255.0

interface LoopBack0
ip address 1.1.1.1 255.255.255.255

bgp 100
router-id 1.1.1.1
peer 2.2.2.2 as-number 100
peer 2.2.2.2 connect-interface LoopBack0
peer 3.3.3.3 as-number 100
peer 3.3.3.3 connect-interface LoopBack0

ipv4-family unicast
undo synchronization
nexthop recursive-lookup route-policy np-by-rp
peer 2.2.2.2 enable
peer 3.3.3.3 enable

ospf 1
area 0.0.0
network 1.1.1.1 0.0.0.0
network 10.1.0.0 0.0.255.255

ip ip-prefix np-by-rp-ip index 10 permit 0.0.0.0 32

route-policy np-by-rp permit node 10
if-match ip-prefix np-by-rp-ip

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0

interface LoopBack0
ip address 2.2.2.2 255.255.255.255

bgp 100
router-id 2.2.2.2
peer 1.1.1.1 as-number 100
peer 1.1.1.1 connect-interface LoopBack0

ipv4-family unicast
undo synchronization
import-route static
peer 1.1.1.1 enable

ospf 1
area 0.0.0
```

```
network 2.2.2.0 0.0.0.0
network 10.1.1.0 0.0.0.255
#
ip route-static 10.20.1.0 24 NULL 0
#
return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.2.2 255.255.255.0

interface LoopBack0
ip address 3.3.3.3 255.255.255.255

bgp 100
router-id 3.3.3.3
peer 1.1.1.1 as-number 100
peer 1.1.1.1 connect-interface LoopBack0

ipv4-family unicast
undo synchronization
import-route static
peer 1.1.1.1 enable

ospf 1
area 0.0.0.0
network 3.3.3.3 0.0.0.0
network 10.1.2.0 0.0.0.255

ip route-static 10.20.1.0 24 NULL 0
#
return
```

## Example for Configuring Routing Policies to Control BGP Route Advertisement and Acceptance

Routing policies can be configured to flexibly filter BGP routes, allowing only desired routes to be advertised and accepted so that these routes guide network traffic forwarding properly.

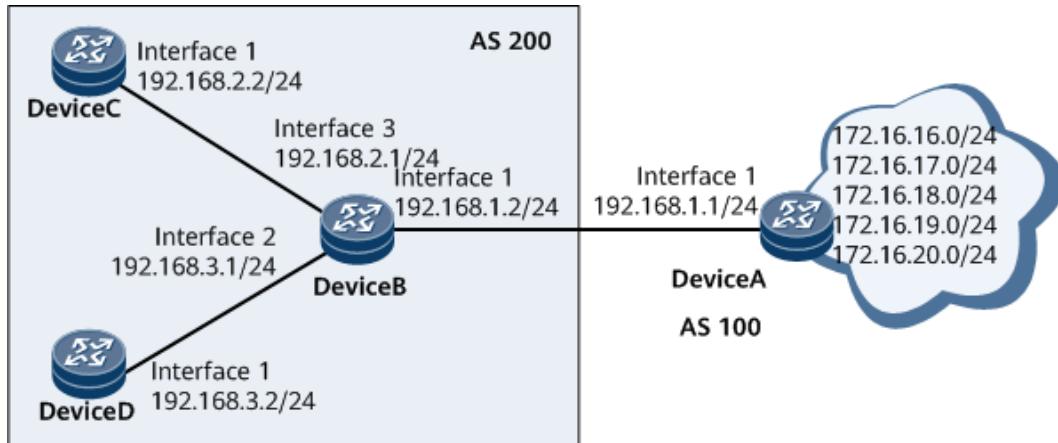
### Networking Requirements

On the network shown in [Figure 1-437](#), DeviceB, DeviceC, and DeviceD reside in AS 200 and run OSPF, whereas DeviceA resides in AS 100. DeviceA receives routes from the Internet. It is required that DeviceA provide only the routes 172.16.17.0/24, 172.16.18.0/24, and 172.16.19.0/24 for DeviceB, DeviceC accept only the route 172.16.18.0/24, and DeviceD accept all the routes provided by DeviceB.

**Figure 1-437** Network diagram of filtering routes to be advertised and accepted

 **NOTE**

Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



## Precautions

When configuring routing policies to control BGP route advertisement and acceptance, note the following:

- When configuring an IP prefix list, specify a proper IP prefix range according to actual requirements.
- Note that the name of a configured IP prefix list is case-sensitive.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on DeviceB, DeviceC, and DeviceD.
2. Establish an EBGP peer relationship between DeviceA and DeviceB. Establish IBGP peer relationships between DeviceB and DeviceC, and between DeviceB and DeviceD.
3. Configure static routes on DeviceA and import them to the BGP routing table.
4. Configure an export routing policy for BGP routes on DeviceA and check the filtering result on DeviceB.
5. Configure an import routing policy for BGP routes on DeviceC and check the filtering result on DeviceC.

## Data Preparation

To complete the configuration, you need the following data:

- Five static routes configured and imported on DeviceA
- OSPF backbone area (area 0) where DeviceB, DeviceC, and DeviceD reside
- Names of IP prefix lists and routes to be filtered

## Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#).

**Step 2** Configure OSPF in AS 100.

# Configure DeviceB.

```
[~DeviceB] ospf
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.2.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.3.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
```

# Configure DeviceC.

```
[~DeviceC] ospf
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 192.168.2.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
[~DeviceC-ospf-1] quit
```

# Configure DeviceD.

```
[~DeviceD] ospf
[*DeviceD-ospf-1] area 0
[*DeviceD-ospf-1-area-0.0.0.0] network 192.168.3.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0.0] commit
[~DeviceD-ospf-1-area-0.0.0.0] quit
```

**Step 3** Configure basic BGP functions.

# Configure DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 192.168.1.2 as-number 200
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 192.168.1.1 as-number 100
[*DeviceB-bgp] peer 192.168.2.2 as-number 200
[*DeviceB-bgp] peer 192.168.3.2 as-number 200
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure DeviceC.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 192.168.2.1 as-number 200
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Configure DeviceD.

```
[~DeviceD] bgp 200
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 192.168.3.1 as-number 200
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

**Step 4** Configure five static routes on DeviceA and import them to the BGP routing table.

```
[~DeviceA] ip route-static 172.16.16.0 24 NULL0
[*DeviceA] ip route-static 172.16.17.0 24 NULL0
[*DeviceA] ip route-static 172.16.18.0 24 NULL0
```

```
[*DeviceA] ip route-static 172.16.19.0 24 NULL0
[*DeviceA] ip route-static 172.16.20.0 24 NULL0
[*DeviceA] bgp 100
[*DeviceA-bgp] import-route static
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Check the BGP routing table of DeviceB. The command output shows that DeviceB has learned the imported static routes.

```
[~DeviceB] display bgp routing-table
```

```
BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

| Total Number of Routes: 5 |             |     |        |         |          |  |
|---------------------------|-------------|-----|--------|---------|----------|--|
| Network                   | NextHop     | MED | LocPrf | PrefVal | Path/Ogn |  |
| *> 172.16.16.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |
| *> 172.16.17.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |
| *> 172.16.18.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |
| *> 172.16.19.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |
| *> 172.16.20.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |

**Step 5** Configure a routing policy to filter the BGP routes to be advertised.

# Configure an IP prefix list named **a2b** on DeviceA.

```
[~DeviceA] ip ip-prefix a2b index 10 permit 172.16.17.0 24
[*DeviceA] ip ip-prefix a2b index 20 permit 172.16.18.0 24
[*DeviceA] ip ip-prefix a2b index 30 permit 172.16.19.0 24
[*DeviceA] commit
```

# Configure an export routing policy based on the IP prefix list **a2b** on DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] filter-policy ip-prefix a2b export static
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Check the BGP routing table of DeviceB. The command output shows that DeviceB has received only the three routes that match the IP prefix list **a2b**.

```
[~DeviceB] display bgp routing-table
```

```
BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

| Total Number of Routes: 3 |             |     |        |         |          |  |
|---------------------------|-------------|-----|--------|---------|----------|--|
| Network                   | NextHop     | MED | LocPrf | PrefVal | Path/Ogn |  |
| *> 172.16.17.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |
| *> 172.16.18.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |
| *> 172.16.19.0/24         | 192.168.1.1 | 0   | 0      | 100?    |          |  |

**Step 6** Configure a routing policy to filter the BGP routes to be accepted.

# Configure an IP prefix list named **in** on DeviceC.

```
[~DeviceC] ip ip-prefix in index 10 permit 172.16.18.0 24
[*DeviceC] commit
```

# Configure an import routing policy based on the IP prefix list **in** on DeviceC.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] filter-policy ip-prefix in import
[*DeviceC-bgp] commit
```

# Check the BGP routing table of DeviceC. The command output shows that DeviceC has accepted only the route that matches the IP prefix list **in**.

```
[~DeviceC] display bgp routing-table
```

```
BGP Local router ID is 3.3.3.3
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network NextHop MED LocPrf PrefVal Path/Ogn
*> 172.16.18.0/24 192.168.1.1 0 0 100?
```

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
bgp 100
router-id 1.1.1.1
peer 192.168.1.2 as-number 200
#
ipv4-family unicast
undo synchronization
filter-policy ip-prefix a2b export static
import-route static
peer 192.168.1.2 enable
#
ip ip-prefix a2b index 10 permit 172.16.17.0 24
ip ip-prefix a2b index 20 permit 172.16.18.0 24
ip ip-prefix a2b index 30 permit 172.16.19.0 24
#
ip route-static 172.16.16.0 255.255.255.0 NULL0
ip route-static 172.16.17.0 255.255.255.0 NULL0
ip route-static 172.16.18.0 255.255.255.0 NULL0
ip route-static 172.16.19.0 255.255.255.0 NULL0
ip route-static 172.16.20.0 255.255.255.0 NULL0
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.3.1 255.255.255.0
```

```

interface GigabitEthernet3/0/0
undo shutdown
ip address 192.168.2.1 255.255.255.0

bgp 200
router-id 2.2.2.2
peer 192.168.1.1 as-number 100
peer 192.168.2.2 as-number 200
peer 192.168.3.2 as-number 200

ipv4-family unicast
undo synchronization
peer 192.168.1.1 enable
peer 192.168.2.2 enable
peer 192.168.3.2 enable

ospf 1
area 0.0.0.0
network 192.168.1.0 0.0.0.255
network 192.168.2.0 0.0.0.255
network 192.168.3.0 0.0.0.255

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0

bgp 200
router-id 3.3.3.3
peer 192.168.2.1 as-number 200

ipv4-family unicast
undo synchronization
filter-policy ip-prefix in import
peer 192.168.2.1 enable

ospf 1
area 0.0.0.0
network 192.168.2.0 0.0.0.255

ip ip-prefix in index 10 permit 172.16.18.0 24

return
```

- DeviceD configuration file

```

sysname DeviceD

interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.3.2 255.255.255.0

bgp 200
router-id 4.4.4.4
peer 192.168.3.1 as-number 200

ipv4-family unicast
undo synchronization
peer 192.168.3.1 enable

ospf 1
area 0.0.0.0
network 192.168.3.0 0.0.0.255
#
```

return

## Example for Configuring BFD for BGP

After BFD for BGP is configured, BFD can fast detect the link fault between BGP peers and notify it to BGP so that service traffic can be transmitted along the backup link.

## Networking Requirements

Voice and video services have high requirements for network reliability and stability. If a fault occurs on a network, quick service recovery is required (within 50 ms). BFD for BGP can meet this requirement.

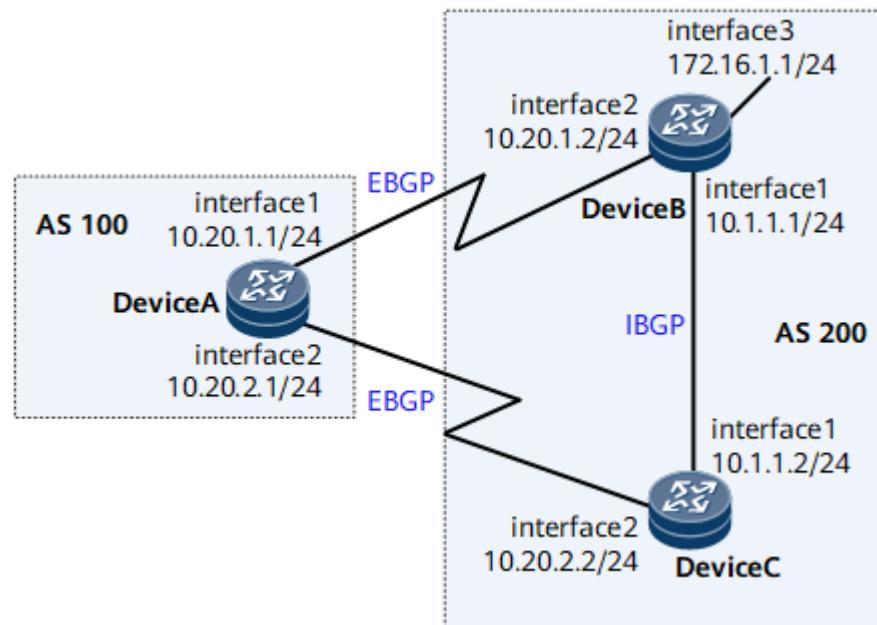
In [Figure 1-438](#), a primary link and a backup link are deployed on the network to ensure service transmission stability. EBGP peer relationships are established between indirectly connected DeviceA and DeviceB, and between indirectly connected DeviceA and DeviceC. In most cases, traffic is transmitted along the primary link between DeviceA and DeviceB. If the primary link fails, it is required that BGP quickly detect this failure and switch traffic to the backup link (DeviceA -> DeviceC -> DeviceB).

BFD for BGP can be configured to speed up the link switchover. Specifically, BFD is configured to track the BGP peer relationship between DeviceA and DeviceB. If the primary link between DeviceA and DeviceB fails, BFD will quickly detect the fault and notify BGP of the fault so that service traffic is switched to the backup link for transmission.

**Figure 1-438** Configuring BFD for BGP



Interfaces 1 through 3 in this example are GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



### NOTE

If two routers establish an EBGP peer relationship over a direct link, BFD for BGP is not required because the **ebgp-interface-sensitive** command is enabled by default for directly connected EBGP peers.

## Precautions

When configuring BFD for BGP, note the following rules:

- Before configuring BFD for BGP, enable BFD globally.
- When configuring BFD for BGP, ensure that parameters configured on the two ends of a BFD session are consistent.
- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic BGP functions on each router.
2. Configure the MED attribute to control route selection.
3. Enable BFD on DeviceA and DeviceB

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of DeviceA, DeviceB, and DeviceC
- Peer IP address to be detected by BFD
- Minimum interval at which BFD Control packets are received and sent and the local detection multiplier

## Procedure

**Step 1** Configure an IP address for each interface on the routers. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic BGP functions, establish EBGP connections between DeviceA and DeviceB, and between DeviceA and DeviceC, and establish an IBGP connection between DeviceB and DeviceC.

# Configure DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.20.1.2 as-number 200
[*DeviceA-bgp] peer 10.20.1.2 ebgp-max-hop 255
[*DeviceA-bgp] peer 10.20.2.2 as-number 200
[*DeviceA-bgp] peer 10.20.2.2 ebgp-max-hop 255
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~DeviceB] bgp 200
```

```
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.20.1.1 as-number 100
[*DeviceB-bgp] peer 10.20.1.1 ebgp-max-hop 255
[*DeviceB-bgp] peer 10.1.1.2 as-number 200
[*DeviceB-bgp] network 172.16.1.0 255.255.255.0
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure DeviceC.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.20.2.1 as-number 100
[*DeviceC-bgp] peer 10.20.2.1 ebgp-max-hop 255
[*DeviceC-bgp] peer 10.1.1.1 as-number 200
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Display peer information on DeviceA. The following command output shows that the BGP peer relationship has been established.

```
<DeviceA> display bgp peer

BGP local router ID : 1.1.1.1
Local AS number : 100
Total number of peers : 2 Peers in established state : 2

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
10.20.1.2 4 200 2 5 0 00:01:25 Established 0
10.20.2.2 4 200 2 4 0 00:00:55 Established 0
```

**Step 3** Configure BFD to detect the BGP peer relationship between DeviceA and DeviceB.

# Enable BFD on DeviceA and establish a BFD session to detect the link between DeviceA and DeviceB.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] commit
```

# Enable BFD on DeviceB and establish a BFD session to detect the link between DeviceB and DeviceA.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] commit
```

**Step 4** Configure the MED attribute.

# Configure a route-policy to set the MED value for the routes that DeviceB and DeviceC send to DeviceA.

# Configure DeviceB.

```
[~DeviceB] route-policy 10 permit node 10
[*DeviceB-route-policy] apply cost 100
[*DeviceB-route-policy] commit
[~DeviceB-route-policy] quit
[~DeviceB] bgp 200
[*DeviceB-bgp] peer 10.20.1.1 route-policy 10 export
[*DeviceB-bgp] commit
```

# Configure DeviceC.

```
[~DeviceC] route-policy 10 permit node 10
[*DeviceC-route-policy] apply cost 150
[*DeviceC-route-policy] commit
```

```
[~DeviceC-route-policy] quit
[~DeviceC] bgp 200
[*DeviceC-bgp] peer 10.20.2.1 route-policy 10 export
[*DeviceC-bgp] commit
```

# Display the BGP routing table of DeviceA.

```
<DeviceA> display bgp routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
Network NextHop MED LocPrf PrefVal Path/Ogn
*> 172.16.1.0/24 10.20.1.2 100 0 200i
* 10.20.2.2 150 0 200i
```

According to the preceding BGP routing table, the next hop address of the route to 172.16.1.0/24 is 10.20.1.2, indicating that traffic is transmitted on the primary link (DeviceA → DeviceB).

**Step 5** Configure BFD, and set the interval at which BFD Control packets are received and sent and the local detection multiplier.

# Enable BFD on DeviceA, set the minimum interval at which BFD Control packets are received and sent to 100 ms, and set the local detection multiplier to 4.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bgp 100
[*DeviceA-bgp] peer 10.20.1.2 bfd enable
[*DeviceA-bgp] peer 10.20.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceA-bgp] commit
```

# Enable BFD on DeviceB, set the minimum interval at which BFD Control packets are received and sent to 100 ms, and set the local detection multiplier to 4.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] bgp 200
[*DeviceB-bgp] peer 10.20.1.1 bfd enable
[*DeviceB-bgp] peer 10.20.1.1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceB-bgp] commit
```

# Check all the BFD sessions established by BGP on DeviceA.

```
<DeviceA> display bgp bfd session all

Local_Address Peer_Address Interface
10.20.1.1 10.20.1.2 GigabitEthernet1/0/0
Tx-interval(ms) Rx-interval(ms) Multiplier Session-State
100 100 4 Up
Wtr-interval(m)
0
```

**Step 6** Verify the configuration.

# Run the **shutdown** command on GE 2/0/0 of DeviceB to simulate a fault on the primary link.

```
[~DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] shutdown
```

```
[*DeviceB-GigabitEthernet2/0/0] commit
Display the BGP routing table of DeviceA.
<DeviceA> display bgp routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network NextHop MED LocPrf PrefVal Path/Ogn
*> 172.16.1.0/24 10.20.2.2 150 0 200i
```

The command output shows that the backup link DeviceA → DeviceC → DeviceB takes effect after the primary link fails and that the next hop address of the route to 172.16.1.0/24 has become 10.20.2.2.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.20.2.1 255.255.255.0

bgp 100
router-id 1.1.1.1
peer 10.20.1.2 as-number 200
peer 10.20.1.2 ebgp-max-hop 255
peer 10.20.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.20.1.2 bfd enable
peer 10.20.2.2 as-number 200
peer 10.20.2.2 ebgp-max-hop 255

ipv4-family unicast
undo synchronization
peer 10.20.1.2 enable
peer 10.20.2.2 enable

return
```

- DeviceB configuration file

```

sysname DeviceB

bfd

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
```

```
ip address 10.20.1.2 255.255.255.0
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
bgp 200
router-id 2.2.2.2
peer 10.1.1.2 as-number 200
peer 10.20.1.1 as-number 100
peer 10.20.1.1 ebgp-max-hop 255
peer 10.20.1.1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.20.1.1 bfd enable
#
ipv4-family unicast
undo synchronization
network 172.16.1.0 255.255.255.0
peer 10.1.1.2 enable
peer 10.20.1.1 enable
peer 10.20.1.1 route-policy 10 export
#
route-policy 10 permit node 10
apply cost 100
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.20.2.2 255.255.255.0
#
bgp 200
router-id 3.3.3.3
peer 10.1.1.1 as-number 200
peer 10.20.2.1 as-number 100
peer 10.20.2.1 ebgp-max-hop 255
#
ipv4-family unicast
undo synchronization
peer 10.1.1.1 enable
peer 10.20.2.1 enable
peer 10.20.2.1 route-policy 10 export
#
route-policy 10 permit node 10
apply cost 150
#
return
```

## Example for Configuring BGP Auto FRR

BGP Auto FRR provides backup forwarding entries for routes, minimizing the delay for important services.

## Networking Requirements

As networks evolve continuously, voice, on-line video, and financial services raise increasingly high requirements for real-time performance. Usually, primary and backup links are deployed on a network to ensure the stability of these services. If

the primary link fails, the device needs to wait for route convergence to be completed. After that, the device reselects an optimal route and delivers this route to the FIB to start a link switchover. This is the traditional switchover mode. In this mode, service interruption lasts a long time, which does not meet the services' requirement.

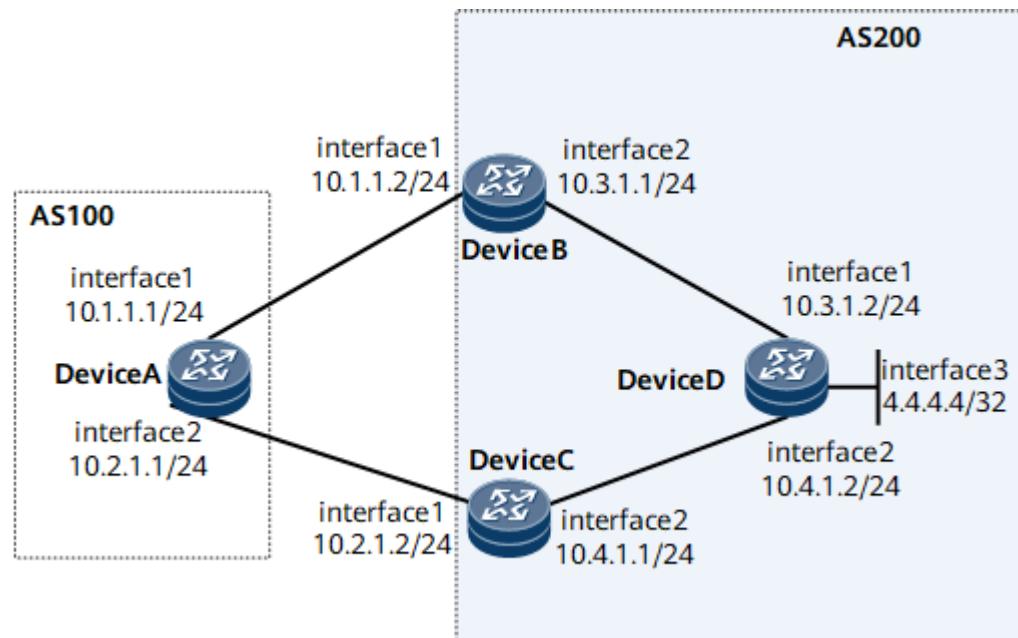
BGP Auto FRR addresses this problem. After BGP Auto FRR is enabled on a device, the device selects the optimal route to forward packets. In addition, the device automatically adds information about the sub-optimal route to the backup forwarding entries of the optimal route and delivers the backup forwarding entries to the FIB. If the primary link fails, the device quickly switches traffic to the backup link. The switchover does not depend on route convergence and reduces service interruption time. The switchover can be performed within sub-seconds.

As shown in **Figure 1-439**, Device A belongs to AS 100; Device B, Device C, and Device D belong to AS 200. BGP Auto FRR needs to be configured to ensure that the route from Device A to DeviceD has the backup route.

**Figure 1-439** Configuring BGP Auto FRR

 **NOTE**

Interfaces 1 through 3 in this example represent GE 1/0/0, GE 2/0/0, and Loopback1, respectively.



## Precautions

When configuring BGP Auto FRR, note the following rules:

- When configuring BGP FRR, ensure that there are at least two routes to the same destination network segment.
- The name of a route-policy is case sensitive.

- To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure EBGP connections between Device A and Device B, and between Device A and Device C. Configure IBGP connections between Device D and Device B, and between Device D and Device C.
2. Configure route-policies on Device B and Device C to change the MED values of routes to Device D for route selection.
3. Configure BGP Auto FRR on Device A.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A, Device B, Device C, and Device D
- Names of route-policies and MED values of routes on Device B and Device C

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure EBGP connections between Device A and Device B, and between Device A and Device C, and configure IBGP connections between Device B and Device D, and between Device C and Device D.

# Configure EBGP connections on Device A.

```
<DeviceA> system-view
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.1.1.2 as-number 200
[*DeviceA-bgp] peer 10.2.1.2 as-number 200
[*DeviceA-bgp] commit
```

### NOTE

The configurations on Device B and Device C are similar to the configuration on Device A.

# Configure IBGP connections on Device D.

```
<DeviceD> system-view
[~DeviceD] bgp 200
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 10.3.1.1 as-number 200
[*DeviceD-bgp] peer 10.4.1.1 as-number 200
[*DeviceD-bgp] commit
```

### NOTE

The configurations on Device B and Device C are similar to the configuration on Device D.

**Step 3** Configure BFD for BGP on Device A, Device B, Device C and Device D.

# Configure BFD for BGP on Device A.

```
<DeviceA> system-view
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bgp 100
[*DeviceA-bgp] peer 10.1.1.2 bfd enable
[*DeviceA-bgp] peer 10.1.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceA-bgp] peer 10.2.1.2 bfd enable
[*DeviceA-bgp] peer 10.2.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
[~DeviceA] quit
```

#### NOTE

The configurations on Device B, Device C and Device D are similar to the configuration on Device A.

- Step 4** Configure route-policies on Device B and Device C to ensure that the MED values of routes to Device D are different.

# Configure a route-policy on Device B.

```
<DeviceB> system-view
[~DeviceB] route-policy rtb permit node 10
[*DeviceB-route-policy] apply cost 80
[*DeviceB-route-policy] quit
[*DeviceB] bgp 200
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] peer 10.1.1.1 route-policy rtb export
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
```

# Configure a route-policy on Device C.

```
<DeviceC> system-view
[~DeviceC] route-policy rtc permit node 10
[*DeviceC-route-policy] apply cost 120
[*DeviceC-route-policy] quit
[*DeviceC] bgp 200
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] peer 10.2.1.1 route-policy rtc export
[*DeviceC-bgp-af-ipv4] commit
[~DeviceC-bgp-af-ipv4] quit
```

# Advertise a route to 4.4.4.4/32 on Device D.

```
[~DeviceD] bgp 200
[*DeviceD-bgp] ipv4-family unicast
[*DeviceD-bgp] network 4.4.4.4 32
[*DeviceD-bgp] commit
```

# Run the **display ip routing-table verbose** command on Device A to check detailed information about the route to 4.4.4.4/32 it learns.

```
<DeviceA> display ip routing-table 4.4.4.4 32 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination: 4.4.4.4/32
 Protocol: EBGP Process ID: 0
 Preference: 255 Cost: 80
 NextHop: 10.1.1.2 Neighbour: 10.1.1.2
 State: Active Adv Age: 00h00m12s
 Tag: 0 Priority: low
 Label: NULL QoSInfo: 0x0
 IndirectID: 0x4
```

```
RelayNextHop: 0.0.0.0 Interface: GigabitEthernet1/0/0
TunnelID: 0x0 Flags: D
```

Because the MED value of the route learned from Device B is smaller, Device A selects the path Device A → Device B → Device D to transmit traffic to 4.4.4.4/32. Because FRR has not been configured yet, no information about the backup route is available.

**Step 5** Enable BGP Auto FRR on Device A, and check the routing information.

```
Enable BGP Auto FRR on Device A.
```

```
<DeviceA> system-view
[~DeviceA] bgp 100
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] auto-frr
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
```

```
Run the display ip routing-table verbose command on Device A to check the routing information.
```

```
<DeviceA> display ip routing-table 4.4.4.4 32 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : _public_
Summary Count : 1

Destination: 4.4.4.4/32
 Protocol: EBGP Process ID: 0
 Preference: 255 Cost: 80
 NextHop: 10.1.1.2 Neighbour: 10.1.1.2
 State: Active Adv Age: 00h52m45s
 Tag: 0 Priority: low
 Label: NULL QoSInfo: 0x0
 IndirectID: 0x4
 RelayNextHop: 0.0.0.0 Interface: GigabitEthernet1/0/0
 TunnelID: 0x0 Flags: D
 BkNextHop: 10.2.1.2 BkInterface: GigabitEthernet2/0/0
 BklLabel: NULL SecTunnelID: 0x0
 BkPETunnelID: 0x0 BkPESecTunnelID: 0x0
 BkIndirectID: 0x2
```

The command output shows that DeviceA has a backup next hop and a backup outbound interface to 4.4.4.4/32.

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0
#
bgp 100
router-id 1.1.1.1
```

```
peer 10.1.1.2 as-number 200
peer 10.1.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.1.1.2 bfd enable
peer 10.2.1.2 as-number 200
peer 10.2.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.2.1.2 bfd enable
#
ipv4-family unicast
undo synchronization
auto-frr
peer 10.1.1.2 enable
peer 10.2.1.2 enable
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.3.1.1 255.255.255.0
#
bgp 200
router-id 2.2.2.2
peer 10.1.1.1 as-number 100
peer 10.1.1.1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.1.1.1 bfd enable
peer 10.3.1.2 as-number 200
peer 10.3.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.3.1.2 bfd enable
#
ipv4-family unicast
undo synchronization
peer 10.1.1.1 route-policy rtb export
peer 10.1.1.1 enable
peer 10.3.1.2 enable
#
route-policy rtb permit node 10
apply cost 80
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.4.1.1 255.255.255.0
#
bgp 200
router-id 3.3.3.3
peer 10.2.1.1 as-number 100
peer 10.2.1.1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.2.1.1 bfd enable
peer 10.4.1.2 as-number 200
peer 10.4.1.2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
```

```
peer 10.4.1.2 bfd enable
#
ipv4-family unicast
undo synchronization
peer 10.2.1.1 route-policy rtc export
peer 10.2.1.1 enable
peer 10.4.1.2 enable
#
route-policy rtc permit node 10
apply cost 120
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
bfd
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.3.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.4.1.2 255.255.255.0
#
interface LoopBack1
ip address 4.4.4.4 255.255.255.255
#
bgp 200
router-id 4.4.4.4
peer 10.3.1.1 as-number 200
peer 10.3.1.1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.3.1.1 bfd enable
peer 10.4.1.1 as-number 200
peer 10.4.1.1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 10.4.1.1 bfd enable
#
ipv4-family unicast
undo synchronization
network 4.4.4.4 255.255.255.255
peer 10.3.1.1 enable
peer 10.4.1.1 enable
#
return
```

## Example for Configuring BGP ADD-PATH

With BGP ADD-PATH, a route reflector (RR) can send two or more routes with the same prefix to a specified IBGP peer. After reaching the IBGP peer, these routes can back up each other or load-balance traffic, which ensures high reliability in data transmission.

## Networking Requirements

In a scenario with an RR and clients, if the RR has multiple routes to the same destination (with the same prefix), the RR selects an optimal route from these routes and then sends only the optimal route to its clients. Therefore, the clients have only one route to the destination. If a link along this route fails, route convergence takes a long time, which cannot meet the requirements for high reliability.

To address this issue, deploy the BGP ADD-PATH feature on the RR. With BGP ADD-PATH, the RR can send two or more routes with the same prefix to a

specified IBGP peer. These routes can back up each other or load-balance traffic, which ensures high reliability in data transmission.

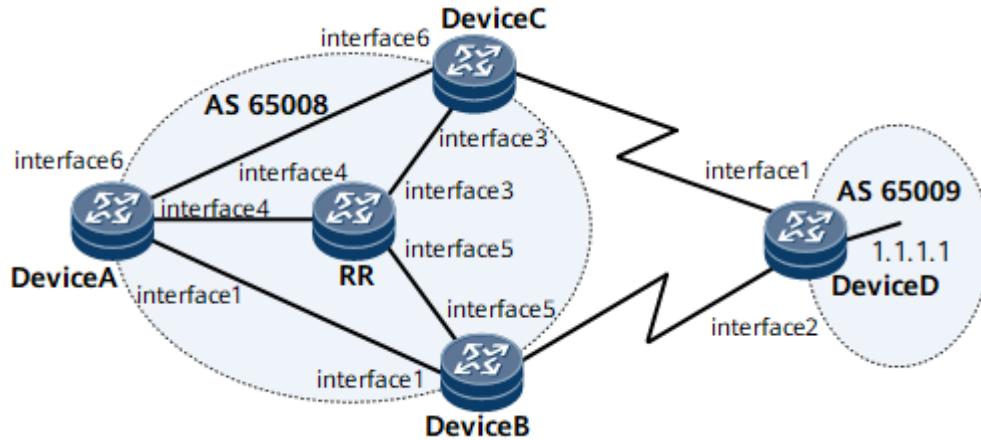
On the network shown in [Figure 1-440](#), Device A, Device B, and Device C are clients of the RR, and Device D is an EBGP peer of Device B and Device C.

To ensure high reliability in data transmission, configure BGP Add-Path on the RR and enable DeviceA to receive Add-Path routes from the RR so that DeviceA can have multiple routes with the same prefix.

**Figure 1-440** Networking for configuring BGP ADD-PATH

 NOTE

Interfaces 1 through 6 in this example represent GE 3/0/0, GE 3/0/2, GE 3/0/3, GE 1/0/1, GE 1/0/2, and GE 1/0/3, respectively.



## Precautions

When configuring the BGP Add-Path function, enable the capability of sending Add-Path routes on the route sender and the capability of receiving Add-Path routes on the route receiver so that Add-Path routes can be exchanged between the two ends.

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an IP address for each interface on each router.
2. Configure basic BGP functions on each router.
3. Enable BGP ADD-PATH on the RR, enable the RR to send ADD-PATH routes to Device A, and configure the number of routes that the RR can send to Device A.

4. Enable Device A to receive BGP ADD-PATH routes from the RR.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs of Device A, Device B, Device C, Device D, and the RR, and their AS numbers, as listed in [Table 1-155](#)

**Table 1-155** Configurations of each device

| Device   | Router ID | Interface             | IP Address    | AS Number |
|----------|-----------|-----------------------|---------------|-----------|
| Device A | 1.1.1.1   | GigabitEthernet 3/0/0 | 172.16.3.1/24 | AS 65008  |
|          |           | GigabitEthernet 1/0/1 | 172.16.2.1/24 |           |
|          |           | GigabitEthernet 1/0/3 | 172.16.1.1/24 |           |
| Device B | 2.2.2.2   | GigabitEthernet 3/0/0 | 172.16.3.2/24 | AS 65008  |
|          |           | GigabitEthernet 3/0/2 | 172.16.7.1/24 |           |
|          |           | GigabitEthernet 1/0/2 | 172.16.5.2/24 |           |
| Device C | 3.3.3.3   | GigabitEthernet 3/0/0 | 172.16.6.1/24 | AS 65008  |
|          |           | GigabitEthernet 3/0/3 | 172.16.4.2/24 |           |
|          |           | GigabitEthernet 1/0/3 | 172.16.1.2/24 |           |
| Device D | 4.4.4.4   | GigabitEthernet 3/0/0 | 172.16.6.2/24 | AS 65009  |
|          |           | GigabitEthernet 3/0/2 | 172.16.7.2/24 |           |
|          |           | LoopBack0             | 1.1.1.1/32    |           |
| RR       | 5.5.5.5   | GigabitEthernet 3/0/3 | 172.16.4.1/24 | AS 65008  |
|          |           | GigabitEthernet 1/0/1 | 172.16.2.2/24 |           |
|          |           | GigabitEthernet 1/0/2 | 172.16.5.1/24 |           |

## Procedure

- Step 1** Configure an IP address for each interface on each router. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic BGP functions. Establish an IBGP peer relationship between DeviceA and the RR, between DeviceB and the RR, and between DeviceC and the RR. Configure DeviceA, DeviceB, and DeviceC as clients of the RR. Establish an EBGP peer relationship between DeviceB and DeviceD, and between DeviceC and DeviceD.

# Configure Device A.

```
[~DeviceA] bgp 65008
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 172.16.2.2 as-number 65008
[*DeviceA-bgp] import-route direct
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 65008
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 172.16.5.1 as-number 65008
[*DeviceB-bgp] peer 172.16.7.2 as-number 65009
[*DeviceB-bgp] import-route direct
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 65008
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 172.16.4.1 as-number 65008
[*DeviceC-bgp] peer 172.16.6.2 as-number 65009
[*DeviceC-bgp] import-route direct
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 65009
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 172.16.6.1 as-number 65008
[*DeviceD-bgp] peer 172.16.7.1 as-number 65008
[*DeviceD-bgp] import-route direct
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

# Configure the RR.

```
[~RR] bgp 65008
[*RR-bgp] router-id 5.5.5.5
[*RR-bgp] peer 172.16.2.1 as-number 65008
[*RR-bgp] peer 172.16.4.2 as-number 65008
[*RR-bgp] peer 172.16.5.2 as-number 65008
[*RR-bgp] peer 172.16.2.1 reflect-client
[*RR-bgp] peer 172.16.4.2 reflect-client
[*RR-bgp] peer 172.16.5.2 reflect-client
[*RR-bgp] import-route direct
[*RR-bgp] commit
[~RR-bgp] quit
```

# Display information about the routes to 1.1.1.1 on Device A.

```
[~DeviceA] display bgp routing-table 1.1.1.1
```

```
BGP local router ID : 1.1.1.1
Local AS number : 65008
Paths: 1 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 1.1.1.1/32:
From: 172.16.2.2 (5.5.5.5)
Route Duration: 0d00h00m25s
Relay IP Nexthop: 172.16.2.2
Relay IP Out-interface: GigabitEthernet1/0/1
Original nexthop: 172.16.7.2
Qos information : 0x0
AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Originator: 2.2.2.2
Cluster list: 5.5.5.5
Not advertised to any peer yet
```

The command output shows that Device A received only one BGP route to 1.1.1.1 from the RR before BGP ADD-PATH is configured.

- Step 3** Enable BGP ADD-PATH on the RR and enable Device A to receive BGP ADD-PATH routes from the RR.

# Configure the RR.

```
[~RR] bgp 65008
[~RR-bgp] bestroute add-path path-number 2
[*RR-bgp] peer 172.16.2.1 capability-advertise add-path send
[*RR-bgp] peer 172.16.2.1 advertise add-path path-number 2
[*RR-bgp] commit
[~RR-bgp] quit
```

# Configure Device A.

```
[~DeviceA] bgp 65008
[~DeviceA-bgp] peer 172.16.2.2 capability-advertise add-path receive
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Display information about the routes to 1.1.1.1 on Device A.

```
[~DeviceA] display bgp routing-table 1.1.1.1

BGP local router ID : 1.1.1.1
Local AS number : 65008
Paths: 2 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 1.1.1.1/32:
From: 172.16.2.2 (5.5.5.5)
Route Duration: 0d00h00m48s
Relay IP Nexthop: 172.16.2.2
Relay IP Out-interface: GigabitEthernet1/0/1
Original nexthop: 172.16.7.2
Qos information : 0x0
AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Received path-id: 0
Originator: 2.2.2.2
Cluster list: 5.5.5.5
Not advertised to any peer yet

BGP routing table entry information of 1.1.1.1/32:
From: 172.16.2.2 (5.5.5.5)
Route Duration: 0d00h00m48s
Relay IP Nexthop: 172.16.2.2
Relay IP Out-interface: GigabitEthernet1/0/1
Original nexthop: 172.16.6.2
Qos information : 0x0
AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, not preferred for router ID
Received path-id: 1
```

```
Originator: 3.3.3.3
Cluster list: 5.5.5.5
Not advertised to any peer yet
```

The command output shows that Device A received two routes from the RR. The route with the original nexthop 172.16.7.2 is the optimal route selected by the RR, and the other one with the original nexthop 172.16.6.2 is an ADD-PATH route.

# Display information about the routes to 1.1.1.1 on the RR.

```
[~RR] display bgp routing-table 1.1.1.1

BGP local router ID : 5.5.5.5
Local AS number : 65008
Paths: 2 available, 1 best, 1 select, 0 best-external, 1 add-path
BGP routing table entry information of 1.1.1.1/32:
RR-client route.
From: 172.16.5.2 (2.2.2.2)
Route Duration: 0d00h19m39s
Relay IP Nexthop: 172.16.5.2
Relay IP Out-interface: GigabitEthernet1/0/2
Original nexthop: 172.16.7.2
Qos information : 0x0
AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Advertised to such 3 peers:
 172.16.5.2
 172.16.4.2
 172.16.2.1

BGP routing table entry information of 1.1.1.1/32:
RR-client route.
From: 172.16.4.2 (3.3.3.3)
Route Duration: 0d00h19m41s
Relay IP Nexthop: 172.16.4.2
Relay IP Out-interface: GigabitEthernet3/0/3
Original nexthop: 172.16.6.2
Qos information : 0x0
AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, add-path, pre 255, not preferred for router ID
Advertised to such 1 peers:
 172.16.2.1
```

The command output shows that the RR sent the optimal route to all its clients but sent the ADD-PATH route only to Device A.

----End

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.3.1 255.255.255.0
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 172.16.2.1 255.255.255.0
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 172.16.1.1 255.255.255.0
#
bgp 65008
```

```
router-id 1.1.1.1
peer 172.16.2.2 as-number 65008
#
ipv4-family unicast
undo synchronization
import-route direct
peer 172.16.2.2 enable
peer 172.16.2.2 capability-advertise add-path receive
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.3.2 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 172.16.7.1 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 172.16.5.2 255.255.255.0
#
bgp 65008
router-id 2.2.2.2
peer 172.16.5.1 as-number 65008
peer 172.16.7.2 as-number 65009
#
ipv4-family unicast
undo synchronization
import-route direct
peer 172.16.5.1 enable
peer 172.16.7.2 enable
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.6.1 255.255.255.0
#
interface GigabitEthernet3/0/3
undo shutdown
ip address 172.16.4.2 255.255.255.0
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 172.16.1.2 255.255.255.0
#
bgp 65008
router-id 3.3.3.3
peer 172.16.4.1 as-number 65008
peer 172.16.6.2 as-number 65009
#
ipv4-family unicast
undo synchronization
import-route direct
peer 172.16.4.1 enable
peer 172.16.6.2 enable
#
return
```

- Device D configuration file

```

sysname DeviceD

interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.6.2 255.255.255.0

interface GigabitEthernet3/0/2
undo shutdown
ip address 172.16.7.2 255.255.255.0

interface LoopBack0
ip address 1.1.1.1 255.255.255.255

bgp 65009
router-id 4.4.4.4
peer 172.16.6.1 as-number 65008
peer 172.16.7.1 as-number 65008

ipv4-family unicast
undo synchronization
import-route direct
peer 172.16.6.1 enable
peer 172.16.7.1 enable

return
```

- RR configuration file

```

sysname RR

interface GigabitEthernet3/0/3
undo shutdown
ip address 172.16.4.1 255.255.255.0

interface GigabitEthernet1/0/1
undo shutdown
ip address 172.16.2.2 255.255.255.0

interface GigabitEthernet1/0/2
undo shutdown
ip address 172.16.5.1 255.255.255.0

bgp 65008
router-id 5.5.5.5
peer 172.16.2.1 as-number 65008
peer 172.16.4.2 as-number 65008
peer 172.16.5.2 as-number 65008

ipv4-family unicast
undo synchronization
import-route direct
bestroute add-path path-number 2
peer 172.16.2.1 enable
peer 172.16.2.1 reflect-client
peer 172.16.2.1 capability-advertise add-path send
peer 172.16.2.1 advertise add-path path-number 2
peer 172.16.4.2 enable
peer 172.16.4.2 reflect-client
peer 172.16.5.2 enable
peer 172.16.5.2 reflect-client

return
```

## Example for Configuring BGP Keychain Authentication

By configuring keychain authentication between BGP peers, you can enhance the security of BGP connections.

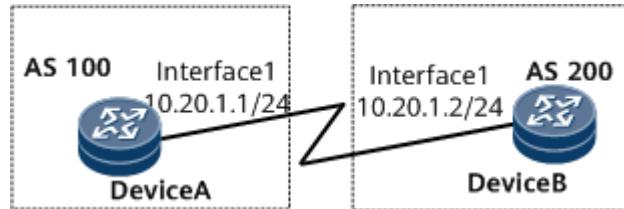
## Networking Requirements

On the network shown in **Figure 1-441**, Device A belongs to AS 100, and Device B belongs to AS 200. BGP runs on the network, and BGP keychain authentication is used to protect EBGP connections against attacks.

**Figure 1-441** Networking diagram of configuring BGP keychain authentication

 NOTE

Interface 1 in this example is GE 1/0/0.



## Precautions

When configuring BGP keychain authentication, pay attention to the following:

- Keychain authentication must be configured on both BGP peers. The keychains configured at both ends must use the same encryption algorithm and password so that a TCP connection can be set up and BGP messages can be exchanged properly.
- For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Establish an EBGP connection between Device A and Device B.
2. Configure keychain authentication on Device A and Device B.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of Device A and Device B
- Name of keychain authentication between Device A and Device B
- Passwords to be encrypted using the HMAC-SHA256 algorithm for the authentication on Device A and Device B

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

**Step 2** Establish an EBGP connection.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.20.1.2 as-number 200
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.20.1.1 as-number 100
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

**Step 3** Configure keychain authentication.

# Configure Device A.

```
[~DeviceA] keychain AMODE mode absolute
[*DeviceA-keychain] tcp-kind 179
[*DeviceA-keychain] tcp-algorithm-id hmac-sha-256 17
[*DeviceA-keychain] receive-tolerance 100
[*DeviceA-keychain] key-id 1
[*DeviceA-keychain-keyid-1] algorithm hmac-sha-256
[*DeviceA-keychain-keyid-1] key-string hello
[*DeviceA-keychain-keyid-1] send-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceA-keychain-keyid-1] receive-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceA-keychain-keyid-1] commit
[~DeviceA-keychain-keyid-1] quit
[~DeviceA-keychain] quit
```

# Configure Device B.

```
[~DeviceB] keychain AMODE mode absolute
[*DeviceB-keychain] tcp-kind 179
[*DeviceB-keychain] tcp-algorithm-id hmac-sha-256 17
[*DeviceB-keychain] receive-tolerance 100
[*DeviceB-keychain] key-id 1
[*DeviceB-keychain-keyid-1] algorithm hmac-sha-256
[*DeviceB-keychain-keyid-1] key-string hello
[*DeviceB-keychain-keyid-1] send-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceB-keychain-keyid-1] receive-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceB-keychain-keyid-1] commit
[~DeviceB-keychain-keyid-1] quit
[~DeviceB-keychain] quit
```

**Step 4** Apply keychain authentication on the EBGP connection between Device A and Device B.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 10.20.1.2 keychain AMODE
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[*DeviceB] bgp 200
```

```
[*DeviceB-bgp] peer 10.20.1.1 keychain AMode
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

**Step 5** Verify the configuration.

# On DeviceA, check the BGP connection status after keychain authentication is configured.

```
<~DeviceA> display bgp peer

BGP local router ID : 10.20.1.1
Local AS number : 100
Total number of peers : 1 Peers in established state : 1

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
10.20.1.2 4 200 21 24 0 00:00:23 Established 0
```

You can view that the status of the BGP connection is Established after keychain authentication is configured.

# Run the **display keychain keychain-name** command on DeviceA to check the Key-id in **Active** state.

```
<~DeviceA> display keychain Amode
Keychain Information:

Keychain Name : amode
Timer Mode : Absolute
Receive Tolerance(min) : 100
Digest Length : 32
Time Zone : LMT
TCP Kind : 179
TCP Algorithm IDs :
 HMAC-MD5 : 5
 HMAC-SHA1-12 : 2
 HMAC-SHA1-20 : 6
 MD5 : 3
 SHA1 : 4
 HMAC-SHA-256 : 17
 SHA-256 : 8
 SM3 : 9
 AES-128-CMAC : 10
 HMAC-SHA-384 : 11
 HMAC-SHA-512 : 12
Number of Key IDs : 1
Active Send Key ID : 1
Active Receive Key IDs : 01
Default send Key ID : Not configured

Key ID Information:

Key ID : 1
Key string : *****
Algorithm : HMAC-SHA-256
SEND TIMER :
 Start time : 2013-08-10 10:00
 End time : 2022-10-28 12:00
 Status : Active
RECEIVE TIMER :
 Start time : 2013-08-10 10:00
 End time : 2022-10-28 12:00
 Status : Active
```

# Run the **display bgp peer ipv4-address verbose** command to verify that the authentication type configured for the BGP peer is **Keychain(AMode)**.

```
<~DeviceA> display bgp ipv6 peer 10.20.1.2 verbose
BGP Peer is 10.20.1.2, remote AS 200
```

```
Type: EBGP link
BGP version 4, Remote router ID 2.2.2.2
Update-group ID: 3
BGP current state: Established, Up for 00h03m36s
BGP current event: KATimerExpired
BGP last state: OpenConfirm
BGP Peer Up count: 1
Received total routes: 0
Received active routes total: 0
Advertised total routes: 0
Port: Local - 179 Remote - 55423
Configured: Connect-retry Time: 32 sec
Configured: Min Hold Time: 0 sec
Configured: Active Hold Time: 180 sec Keepalive Time:60 sec
Received : Active Hold Time: 180 sec
Negotiated: Active Hold Time: 180 sec Keepalive Time:60 sec
Peer optional capabilities:
Peer supports bgp multi-protocol extension
Peer supports bgp route refresh capability
Peer supports bgp 4-byte-as capability
Address family IPv4 Unicast: advertised and received
Received: Total 6 messages
 Update messages 1
 Open messages 1
 KeepAlive messages 4
 Notification messages 0
 Refresh messages 0
Sent: Total 7 messages
 Update messages 1
 Open messages 1
 KeepAlive messages 5
 Notification messages 0
 Refresh messages 0
Authentication type configured: Keychain(amode)
Last keepalive received: 2013-08-15 17:51:17+00:00
Last keepalive sent : 2013-08-15 17:51:52+00:00
Last update received: 2013-08-15 17:48:27+00:00
Last update sent : 2013-08-15 17:48:27+00:00
No refresh received since peer has been configured
No refresh sent since peer has been configured
Minimum route advertisement interval is 30 seconds
Optional capabilities:
Route refresh capability has been enabled
4-byte-as capability has been enabled
Peer Preferred Value: 0
Memory Priority: medium
Routing policy configured:
No routing policy is configured
```

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

keychain AMode mode absolute
receive-tolerance 100
tcp-kind 179
tcp-algorithm-id hmac-sha-256 17

key-id 1
algorithm hmac-sha-256
key-string cipher %#%#e^1}%%w;/C[M)OQc7"j+,2)}%#%#
send-time 11:00 2009-12-24 to 12:00 2009-12-24
receive-time 11:00 2009-12-24 to 12:00 2009-12-24
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.1.1 255.255.255.0
#
bgp 100
router-id 1.1.1.1
peer 10.20.1.2 as-number 200
peer 10.20.1.2 keychain AMode
#
ipv4-family unicast
undo synchronization
peer 10.20.1.2 enable
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
keychain AMode mode absolute
receive-tolerance 100
tcp-kind 179
tcp-algorithm-id hmac-sha-256 17
#
key-id 1
algorithm hmac-sha-256
key-string cipher %#%#ub(70WJ"^-i(kxPK@*fK,){t%#%#
send-time 11:00 2009-12-24 to 12:00 2009-12-24
receive-time 11:00 2009-12-24 to 12:00 2009-12-24
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.20.1.2 255.255.255.0
#
bgp 200
router-id 2.2.2.2
peer 10.20.1.1 as-number 100
peer 10.20.1.1 keychain AMode
#
ipv4-family unicast
undo synchronization
peer 10.20.1.1 enable
#
return
```

## Example for Configuring BGP-LS

BGP-link state (LS) enables BGP to report topology information collected by IGPs to the controller.

## Networking Requirements

BGP-LS is a new method of collecting topology information. With powerful routing capabilities of BGP, BGP-LS has the following advantages:

- Reduces computing capability requirements and spares the necessity of IGPs on the controller.
- Facilitates route selection and calculation on the controller by using BGP to summarize process or AS topology information and report the complete information to the controller.
- Requires only one routing protocol (BGP) to report topology information to the controller.

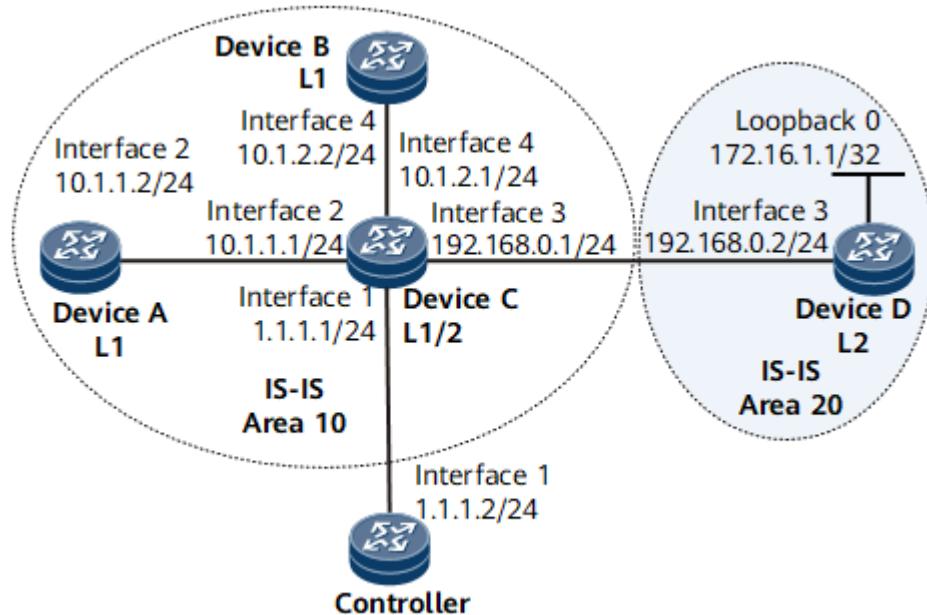
In [Figure 1-442](#), Device C is connected to the controller and reports topology information to the controller. Device A, Device B, Device C, and Device D use IS-IS

to communicate with each other at the network layer. Device A, Device B, and Device C reside in area 10, whereas Device D resides in area 20. Device A and Device B are Level-1 devices, Device C is a Level-1-2 device, and Device D is a Level-2 device.

**Figure 1-442 Configuring BGP-LS**

 **NOTE**

interface1, interface2, interface3, and interface4 in this example represent GigabitEthernet1/0/1, GigabitEthernet1/0/2, GigabitEthernet1/0/3, and GigabitEthernet1/0/4, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an IP address for each interface on each router.
2. Configure basic IS-IS functions.
3. Deploy BGP-LS on Device C and the controller.

## Data Preparation

To complete the configuration, you need the following data:

- Area addresses of Device A, Device B, Device C, and Device D
- Levels of Device A, Device B, Device C, and Device D

- BGP-LS identifier of Device C in IS-IS
- BGP AS numbers, BGP-LS domain AS numbers, and BGP-LS domain IDs of Device C and the controller

## Procedure

**Step 1** Assign an IP address to each interface on each router. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic IS-IS functions.

# Configure Device A.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-1
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/2
[*DeviceA-GigabitEthernet1/0/2] isis enable 1
[*DeviceA-GigabitEthernet1/0/2] commit
[~DeviceA-GigabitEthernet1/0/2] quit
```

# Configure DeviceB.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] is-level level-1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 1/0/4
[*DeviceB-GigabitEthernet1/0/4] isis enable 1
[*DeviceB-GigabitEthernet1/0/4] commit
[~DeviceB-GigabitEthernet1/0/4] quit
```

# Configure DeviceC.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/2
[*DeviceC-GigabitEthernet1/0/2] isis enable 1
[*DeviceC-GigabitEthernet1/0/2] quit
[*DeviceC] interface gigabitethernet 1/0/3
[*DeviceC-GigabitEthernet1/0/3] isis enable 1
[*DeviceC-GigabitEthernet1/0/3] quit
[*DeviceC] interface gigabitethernet 1/0/4
[*DeviceC-GigabitEthernet1/0/4] isis enable 1
[*DeviceC-GigabitEthernet1/0/4] commit
[~DeviceC-GigabitEthernet1/0/4] quit
```

# Configure DeviceD.

```
[~DeviceD] isis 1
[*DeviceD-isis-1] is-level level-2
[*DeviceD-isis-1] network-entity 20.0000.0000.0004.00
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 1/0/3
[*DeviceD-GigabitEthernet1/0/3] isis enable 1
[*DeviceD-GigabitEthernet1/0/3] quit
[*DeviceD] interface LoopBack0
[*DeviceD-LoopBack0] isis enable 1
[*DeviceD-LoopBack0] commit
[~DeviceD-LoopBack0] quit
```

# Check IS-IS routing information on each router. The following example uses the command output on Device C.

```
[~DeviceC] display isis route
 Route information for ISIS(1)

 ISIS(1) Level-1 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 1.1.1.0/24 10 NULL GE1/0/1 Direct D/-L/-
 10.1.1.0/24 10 NULL GE1/0/2 Direct D/-L/-
 10.1.2.0/24 10 NULL GE1/0/4 Direct D/-L/-
 192.168.0.0/24 10 NULL GE1/0/3 Direct D/-L/-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set

 ISIS(1) Level-2 Forwarding Table

 IPV4 Destination IntCost ExtCost ExitInterface NextHop Flags

 1.1.1.0/24 10 NULL GE1/0/1 Direct D/-L/-
 10.1.1.0/24 10 NULL GE1/0/2 Direct D/-L/-
 10.1.2.0/24 10 NULL GE1/0/4 Direct D/-L/-
 172.16.1.1/32 10 NULL GE1/0/3 192.168.0.2 A/-/-/
 192.168.0.0/24 10 NULL GE1/0/3 Direct D/-L/-
 Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,
 U-Up/Down Bit Set
```

**Step 3** Deploy BGP-LS on Device C and the controller.

```
Enable IS-IS topology advertisement to BGP on Device C.
```

```
[~DeviceC] isis 1
[*DeviceC-isis-1] bgp-ls enable level-1-2
[*DeviceC-isis-1] bgp-ls identifier 20
[*DeviceC-isis-1] commit
[~DeviceC-isis-1] quit
```

```
Enable BGP-LS on Device C and configure the controller as a BGP-LS peer of
Device C.
```

```
[~DeviceC] bgp 100
[*DeviceC-bgp] peer 1.1.1.2 as-number 100
[*DeviceC-bgp] link-state-family unicast
[*DeviceC-bgp-af-ls] peer 1.1.1.2 enable
[*DeviceC-bgp-af-ls] commit
[~DeviceC-bgp-af-ls] quit
[~DeviceC-bgp] quit
```

```
Enable BGP-LS on the controller and configure Device C as a BGP-LS peer of the
controller.
```

```
[~Controller] bgp 100
[*Controller-bgp] peer 1.1.1.1 as-number 100
[*Controller-bgp] link-state-family unicast
[*Controller-bgp-af-ls] peer 1.1.1.1 enable
[*Controller-bgp-af-ls] commit
[~Controller-bgp-af-ls] quit
[~Controller-bgp] quit
```

**Step 4** Verify the configuration.

```
Display information about BGP-LS peers and their status on Device C.
```

```
[~DeviceC] display bgp link-state unicast peer
BGP local router ID : 10.1.1.1
Local AS number : 100
```

| Total number of peers : 1 |   |     |         |         |      |          | Peers in established state : 1 |         |  |
|---------------------------|---|-----|---------|---------|------|----------|--------------------------------|---------|--|
| Peer                      | V | AS  | MsgRcvd | MsgSent | OutQ | Up/Down  | State                          | PrefRcv |  |
| 1.1.1.2                   | 4 | 100 | 27      | 48      | 0    | 00:29:11 | Established                    | 17      |  |

# Display BGP-LS routes on Device C.

```
[~DeviceC] display bgp link-state unicast routing-table
BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete

Total Number of Node Routes: 6
*> Network : [NODE][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0002.00]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [NODE][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [NODE][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [NODE][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [NODE][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.02]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [NODE][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.01]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?

Total Number of Link Routes: 8
*> Network : [LINK][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0002.00]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.02]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.02]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.02]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0002.00]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
```

```
Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.01]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.01]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.01]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [LINK][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.01]][REMOTE[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]][LINK[if-address0.0.0.0][peer-address0.0.0.0][if-address::][peer-address::]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
Total Number of IPv4 Prefix Routes: 11
*> Network : [IPV4-PREFIX][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0002.00]][PREFIX[ospf-route-type0][prefix10.1.2.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix1.1.1.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix10.1.1.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix10.1.2.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-1][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix192.168.0.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix1.1.1.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix10.1.1.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix10.1.2.0/24]]
 NextHop : 0.0.0.0
 MED : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix192.168.0.0/24]]
```

```
NextHop : 0.0.0.0 LocPrf :
MED : 0 PrefVal : 0
Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]][PREFIX[ospf-route-type0][prefix192.168.0.0/24]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
*> Network : [IPV4-PREFIX][ISIS-LEVEL-2][IDENTIFIER20][LOCAL[as100][bgp-ls-identifier10.1.1.1][ospf-area-id0.0.0.0][igp-router-id0000.0000.0004.00]][PREFIX[ospf-route-type0][prefix172.16.1.1/32]]
 NextHop : 0.0.0.0 LocPrf :
 MED : 0 PrefVal : 0
 Path/Ogn : ?
```

The preceding command output shows that Device C obtains the topology information on the whole IS-IS network. Device C can use BGP-LS routes to report the topology information to its BGP-LS peer (the controller).

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

isis 1
is-level level-1
network-entity 10.0000.0000.0001.00

interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.1.2 255.255.255.0
isis enable 1

return
```

- Device B configuration file

```

sysname DeviceB

isis 1
is-level level-1
network-entity 10.0000.0000.0002.00

interface GigabitEthernet1/0/4
undo shutdown
ip address 10.1.2.2 255.255.255.0
isis enable 1

return
```

- Device C configuration file

```

sysname DeviceC

isis 1
bgp-ls enable level-1-2
bgp-ls identifier 20
network-entity 10.0000.0000.0003.00

interface GigabitEthernet1/0/1
undo shutdown
ip address 1.1.1.1 255.255.255.0

interface GigabitEthernet1/0/2
undo shutdown
```

```
ip address 10.1.1.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 192.168.0.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet1/0/4
undo shutdown
ip address 10.1.2.1 255.255.255.0
isis enable 1
#
bgp 100
peer 1.1.1.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.1.2 enable
#
link-state-family unicast
peer 1.1.1.2 enable
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
isis 1
is-level level-2
network-entity 20.0000.0000.0004.00
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 192.168.0.2 255.255.255.0
isis enable 1
#
interface LoopBack0
ip address 172.16.1.1 255.255.255.255
isis enable 1
#
return
```

- Controller configuration file

```
#
sysname Controller
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 1.1.1.2 255.255.255.0
#
bgp 100
peer 1.1.1.1 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.1.1 enable
#
link-state-family unicast
peer 1.1.1.1 enable
#
return
```

## Example for Configuring BGP RPD

BGP RPD ensures that route-policies are distributed dynamically.

## Networking Requirements

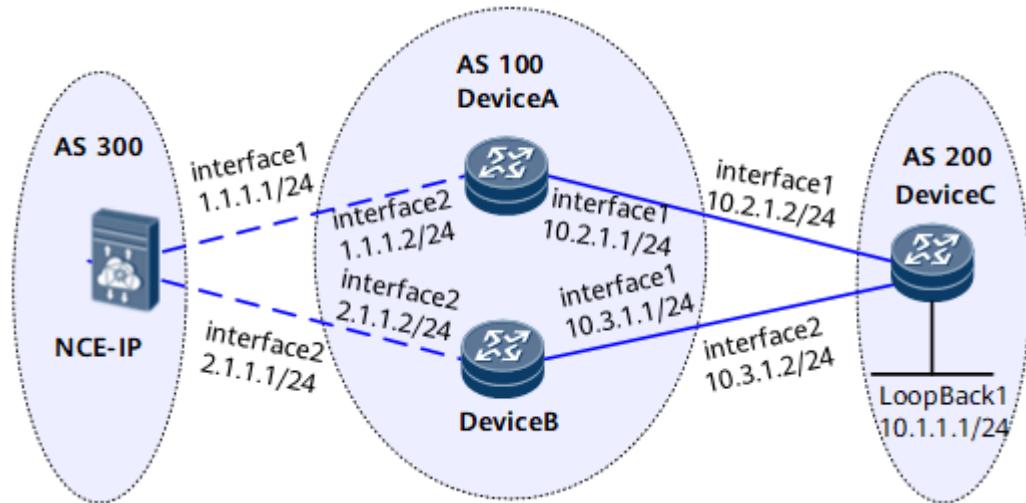
In a MAN ingress or IGW scenario, uneven link resource utilization or link faults may cause link congestion. To fully use network bandwidth, you can deploy an inbound traffic optimization solution to adjust route priorities so that traffic is diverted to idle links. In such a scenario, the router functions as a forwarder, and RPD needs to be deployed on it.

In [Figure 1-443](#), BGP runs on all routers. router A and router B reside in AS 100, router C in AS 200, and the controller in AS 300. The traffic that Device C in AS 200 sends to the destination IP address 192.168.1.0 can enter AS 100 through router A or router B. However, the controller finds that the link between router A and router C is congested. In this case, a traffic optimization policy can be configured so that an RPD route is delivered to divert the traffic to router B when the traffic enters AS 100.

**Figure 1-443** Network diagram of configuring BGP RPD

 NOTE

Interface 1 and interface 2 in this example stand for GE 1/0/0 and GE 1/0/1, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure EBGP connections between Device A and Device C, and between Device B and Device C.
2. On Device A and Device B, enable RPD and establish RPD peer relationships with the controller.

3. Configure IPv4 unicast on Device A, Device B, and Device C so that IPv4 unicast peer relationships are established between Device A and Device C, and between Device B and Device C.

 NOTE

This section provides only the configurations and procedures for forwarders. The controller's configurations, such as the BGP, RPD address family, and traffic optimization policy configurations, are not provided here.

## Data Preparation

To complete the configuration, you need the following data:

- Router ID (4.1.1.1) of Device A, router ID (2.2.2.2) of Device B, and their AS number 100
- Router ID 3.3.3.3 of Device C and its AS number 200

## Procedure

**Step 1** Assign an IP address to each interface. For configuration details, see Configuration Files in this section.

**Step 2** Configure BGP connections.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.2.1.2 as-number 200
[*DeviceA-bgp] peer 1.1.1.1 as-number 300
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] network 192.168.1.0 255.255.255.0
[*DeviceA-bgp-af-ipv4] commit
[~DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 10.3.1.2 as-number 200
[*DeviceB-bgp] peer 2.1.1.1 as-number 300
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] network 192.168.1.0 255.255.255.0
[*DeviceB-bgp-af-ipv4] commit
[~DeviceB-bgp-af-ipv4] quit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 10.2.1.1 as-number 100
[*DeviceC-bgp] peer 10.3.1.1 as-number 100
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] commit
[~DeviceC-bgp-af-ipv4] quit
[~DeviceC-bgp] quit
```

# Check the routing table of Device C.

```
[~DeviceC] display bgp routing-table 192.168.1.0 24
```

```
BGP local router ID : 3.3.3.3
Local AS number : 200
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 192.168.1.0/24:
From: 10.2.1.1 (1.1.1.1)
Route Duration: 0d00h00m56s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.2.1.1
Qos information : 0x0
AS-path 100, origin igp, MED 0, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
 10.2.1.1
 10.3.1.1
```

```
BGP routing table entry information of 192.168.1.0/24:
From: 10.3.1.1 (2.2.2.2)
Route Duration: 0d00h00m06s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.3.1.1
Qos information : 0x0
AS-path 100, origin igp, MED 0, pref-val 0, valid, external, pre 255, not preferred for router ID
Not advertised to any peers yet
```

The preceding command output shows that there are two valid routes to 192.168.1.0/24. The route with the next-hop address of 10.2.1.1 is the optimal route because the router ID of Device A is smaller. In this case, traffic enters AS 100 through Device A.

- Step 3** Configure BGP RPD functions on forwarders so that the forwarders receive RPD routes delivered by the controller and execute corresponding route-policies.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 1.1.1.1 as-number 300
[*DeviceA-bgp] rpd-family
[*DeviceA-bgp-af-rpd] peer 1.1.1.1 enable
[*DeviceA-bgp-af-rpd] quit
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 10.2.1.2 rpd-policy export enable
[*DeviceA-bgp-af-ipv4] commit
[*DeviceA-bgp-af-ipv4] quit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] peer 2.1.1.1 as-number 300
[*DeviceB-bgp] rpd-family
[*DeviceB-bgp-af-rpd] peer 2.1.1.1 enable
[*DeviceB-bgp-af-rpd] quit
[*DeviceB-bgp] ipv4-family unicast
[*DeviceB-bgp-af-ipv4] peer 10.3.1.2 rpd-policy export enable
[*DeviceB-bgp-af-ipv4] commit
[*DeviceB-bgp-af-ipv4] quit
[~DeviceB-bgp] quit
```

# Check information about RPD routes on Device A.

```
[~DeviceA] display bgp rpd routing-table
```

```
Total number of Routes : 1
BGP Local router ID is 4.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
Network Peer MED LocPrf PrefVal Path/Ogn
*> 1/10.2.1.2/1 1.1.1.1 50 0 100?
```

# Check the routing table of Device C.

```
[~DeviceC] display bgp routing-table 192.168.1.0 24

BGP local router ID : 3.3.3.3
Local AS number : 200
Paths: 2 available, 1 best, 1 select
BGP routing table entry information of 192.168.1.0/24:
From: 10.3.1.1 (2.2.2.2)
Route Duration: 0d00h00m06s
Direct Out-interface: GigabitEthernet1/0/1
Original nexthop: 10.3.1.1
Qos information : 0x0
AS-path 100, origin igp, MED 0, pref-val 0, valid, external, best, select, pre 255
Advertised to such 2 peers:
 10.2.1.1
 10.3.1.1

BGP routing table entry information of 192.168.1.0/24:
From: 10.2.1.1 (1.1.1.1)
Route Duration: 0d00h00m56s
Direct Out-interface: GigabitEthernet1/0/0
Original nexthop: 10.2.1.1
Qos information : 0x0
AS-path 100, origin igp, MED 50, pref-val 0, valid, external, pre 255, not preferred for MED
Not advertised to any peers yet
```

The preceding command output shows that the route with the next hop of 10.3.1.1 (Device B) is selected by Device C as the optimal route because its MED value 0 is smaller than that (50) of the route with the next hop of 10.2.1.1 (Device A). In this case, the traffic bypasses the congested link.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.1 255.255.255.0

interface GigabitEthernet1/0/1
undo shutdown
ip address 1.1.1.2 255.255.255.0

bgp 100
router-id 1.1.1.1
peer 1.1.1.1 as-number 300
peer 10.2.1.2 as-number 200

ipv4-family unicast
network 192.168.1.0 255.255.255.0
peer 10.2.1.2 enable
peer 10.2.1.2 rpd-policy export enable

rpd-family
peer 1.1.1.1 enable

return
```

- Device B configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
```

```
ip address 10.3.1.1 255.255.255.0
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 2.1.1.2 255.255.255.0
#
bgp 100
router-id 2.2.2.2
peer 2.1.1.1 as-number 300
peer 10.3.1.2 as-number 200
#
ipv4-family unicast
network 192.168.1.0 255.255.255.0
peer 10.3.1.2 enable
peer 10.3.1.2 rpd-policy export enable
#
rpd-family
peer 2.1.1.1 enable
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.2.1.2 255.255.255.0
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.3.1.2 255.255.255.0
#
bgp 200
router-id 3.3.3.3
peer 10.2.1.1 as-number 100
peer 10.3.1.1 as-number 100
#
ipv4-family unicast
peer 10.2.1.1 enable
peer 10.3.1.1 enable
#
return
```

## Example for Configuring Dynamic BGP Peer Groups

Configuring dynamic BGP peer groups reduces network maintenance workload.

## Networking Requirements

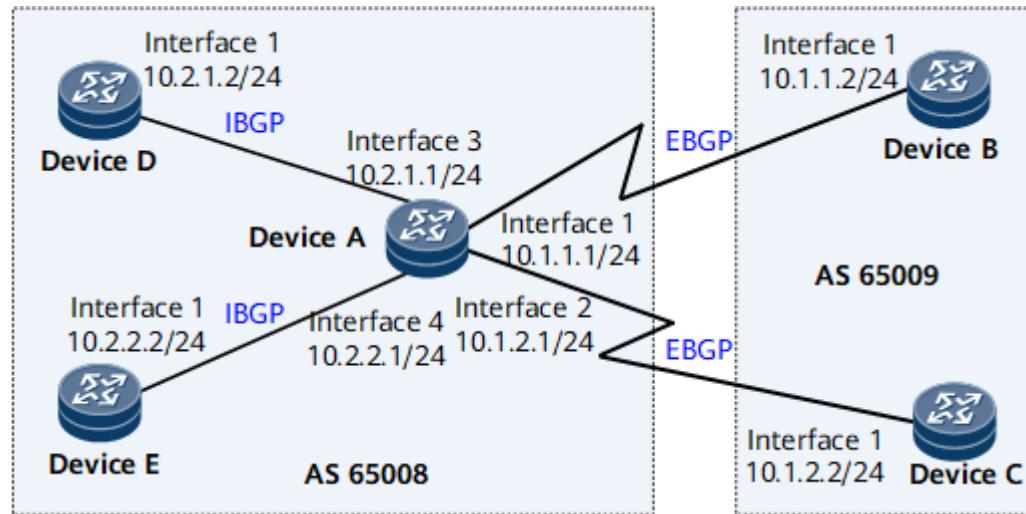
On a BGP network, multiple peers may frequently change, causing the establishment of peer relationships to change accordingly. If you configure peers in static mode, you need to frequently add or delete peer configurations on the local device, which increases the maintenance workload. To address this problem, configure the dynamic BGP peer function to enable BGP to listen for BGP connection requests from a specified network segment, dynamically establish BGP peer relationships, and add these peers to the same dynamic peer group. This spares you from adding or deleting BGP peer configurations in response to each change in BGP peers.

On the network shown in [Figure 1-444](#), DeviceA, DeviceD, and DeviceE are in AS 65008, and DeviceB and DeviceC are in AS 65009. Because many devices are on the same network segment in an AS, you can configure dynamic peer groups on DeviceA.

**Figure 1-444** Network diagram of configuring dynamic BGP peer groups

 NOTE

In this example, interface 1, interface 2, interface 3, and interface 4 represent GE1/0/1, GE1/0/2, GE1/0/3, and GE1/0/4, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure a dynamic IBGP peer group on DeviceA to listen for BGP connection requests from network segment 10.2.0.0/16.
2. Configure a dynamic EBGP peer group on DeviceA to listen for BGP connection requests from network segment 10.1.0.0/16.
3. Configure IBGP connections between DeviceD and DeviceA, and between DeviceE and DeviceA.
4. Configure EBGP connections between DeviceB and DeviceA, and between DeviceC and DeviceA.

## Data Preparation

To complete the configuration, you need the following data:

- AS numbers of DeviceA, DeviceD, and DeviceE
- AS numbers of DeviceB and DeviceC

## Procedure

- Step 1** Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure dynamic BGP peer groups.

```
Configure dynamic BGP peer groups on DeviceA.
```

```
[~DeviceA] bgp 65008
[*DeviceA-bgp] bgp dynamic-session-limit 5
[*DeviceA-bgp] group a listen internal
[*DeviceA-bgp] peer a listen-net 10.2.0.0 16
[*DeviceA-bgp] group b listen external
[*DeviceA-bgp] peer b listen-as 65009
[*DeviceA-bgp] peer b listen-net 10.1.0.0 16
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

**Step 3** Configure IBGP connections.

```
Configure DeviceD.
```

```
[~DeviceD] bgp 65008
[*DeviceD-bgp] peer 10.2.1.1 as-number 65008
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

```
Configure DeviceE.
```

```
[~DeviceE] bgp 65008
[*DeviceE-bgp] peer 10.2.2.1 as-number 65008
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

**Step 4** Configure EBGP connections.

```
Configure DeviceB.
```

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] peer 10.1.1.1 as-number 65008
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

```
Configure DeviceC.
```

```
[~DeviceC] bgp 65009
[*DeviceC-bgp] peer 10.1.2.1 as-number 65008
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

```
Check the status of BGP connections.
```

```
[~DeviceA] display bgp peer
Status codes: * - Dynamic
BGP local router ID : 10.1.1.1
Local AS number : 65008
Total number of peers : 4 Peers in established state : 4
Total number of dynamic peers : 4

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
*10.1.1.2 4 65009 8 7 0 00:04:16 Established 0
*10.1.2.2 4 65009 5 5 0 00:02:01 Established 0
*10.2.1.2 4 65008 5 5 0 00:02:01 Established 0
*10.2.2.2 4 65008 5 5 0 00:02:01 Established 0
```

The command output shows that DeviceA has established BGP connections with other routers and the connection status is **Established**.

```
Check information about BGP peer groups.
```

```
[~DeviceA] display bgp group a
BGP peer-group : a
Remote AS : 65008
```

```

Authentication type configured : None
Type : listen internal
Configured hold timer value : 180
Keepalive timer value : 60
Connect-retry timer value : 32
Minimum route advertisement interval is 15 seconds
PeerSession Members :
 10.2.1.2 10.2.2.2

Peer Preferred Value: 0
No routing policy is configured
Peer Members:
 Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
 *10.2.1.2 4 65008 3 4 0 00:00:20 Established 0
 *10.2.2.2 4 65008 3 4 0 00:00:21 Established 0

[~DeviceA] display bgp group b
BGP peer-group : b
Remote AS : 65009
Authentication type configured : None
Type : listen external
Configured hold timer value : 180
Keepalive timer value : 60
Connect-retry timer value : 32
Minimum route advertisement interval is 15 seconds
PeerSession Members :
 10.1.1.2 10.1.2.2

Peer Preferred Value: 0
No routing policy is configured
Peer Members:
 Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
 *10.1.1.2 4 65009 3 4 0 00:00:20 Established 0
 *10.1.2.2 4 65009 3 4 0 00:00:21 Established 0

```

The command output shows that two dynamic peer groups **a** and **b** exist on DeviceA and each dynamic peer group has two peers, indicating that the dynamic peer groups are properly established.

----End

## Configuration Files

- DeviceA configuration file

```

#
sysname DeviceA
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 10.2.1.1 255.255.255.0
interface GigabitEthernet1/0/4
undo shutdown
ip address 10.2.2.1 255.255.255.0
#
bgp 65008
bgp dynamic-session-limit 5
group a listen internal
peer a listen-net 10.2.0.0 255.255.0.0
group b listen external
peer b listen-as 65009
peer b listen-net 10.1.0.0 255.255.0.0

```

```

ipv4-family unicast
peer a enable
peer b enable

return
```

- DeviceB configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.1.2 255.255.255.0

bgp 65009
peer 10.1.1.1 as-number 65008

ipv4-family unicast
peer 10.1.1.1 enable

return
```

- DeviceC configuration file

```

sysname DeviceC

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.2.2 255.255.255.0

bgp 65009
peer 10.1.2.1 as-number 65008

ipv4-family unicast
peer 10.1.2.1 enable

return
```

- DeviceD configuration file

```

sysname DeviceD

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.2.1.2 255.255.255.0

bgp 65008
peer 10.2.1.1 as-number 65008

ipv4-family unicast
peer 10.2.1.1 enable

return
```

- DeviceE configuration file

```

sysname DeviceE

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.2.2.2 255.255.255.0

bgp 65008
peer 10.2.2.1 as-number 65008

ipv4-family unicast
peer 10.2.2.1 enable
#
```

return

## Example for Configuring BGP Multi-Instance

This section provides an example for configuring BGP multi-instance to achieve instance-specific management and maintenance of routes.

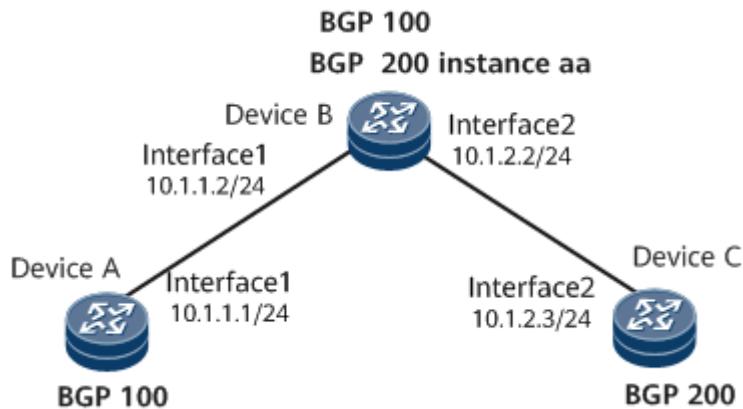
## Networking Requirements

On the network shown in [Figure 1-445](#), the public network BGP-IPv4 unicast address family is enabled in base BGP instances on DeviceA and DeviceB, and an EBGP peer relationship is established to transmit public network routes. In addition, the VPN address family is enabled in the BGP multi-instance on DeviceB and in the base BGP instance on DeviceC, and an EBGP-VPN peer relationship is established to transmit VPN routes. In this way, routes can be managed and maintained separately.

**Figure 1-445** BGP multi-instance networking

 NOTE

In this example, interfaces 1 and 2 represent GE 1/0/0 and GE 2/0/0, respectively.



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Enable the public network BGP-IPv4 unicast address family in base BGP instances on DeviceA and DeviceB, and establish an EBGP peer relationship in this address family.
2. Enable the VPN address family in the BGP multi-instance on DeviceB and in the base instance on DeviceC, and establish an EBGP-VPN peer relationship in this address family.

## Data Preparation

To complete the configuration, you need the following data:

- DeviceA's AS number: 100
- DeviceB's AS numbers: 100 and 200
- DeviceC's AS number: 200

## Procedure

**Step 1** Enable the public network BGP-IPv4 unicast address family in base BGP instances on DeviceA and DeviceB, and establish an EBGP peer relationship in this address family.

# Configure DeviceA.

```
[~HUAWEI] sysname DeviceA
[*HUAWEI] commit
[*DeviceA] interface gigabitethernet1/0/0
[*DeviceA-GigabitEthernet1/0/0] ip address 10.1.1.1 24
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 10.1.1.2 as-number 100
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[*DeviceB] interface gigabitethernet1/0/0
[*DeviceB-GigabitEthernet1/0/0] ip address 10.1.1.2 24
[*DeviceB-GigabitEthernet1/0/0] commit
[~DeviceB-GigabitEthernet1/0/0] quit
[~DeviceB] bgp 100
[*DeviceB-bgp] peer 10.1.1.1 as-number 100
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

After the configuration is complete, check the connectivity between DeviceA and DeviceB. Then, run the **display bgp peer** command. The command output shows that the public network EBGP peer relationship between DeviceA and DeviceB is **Established**.

DeviceB is used as an example.

```
<DeviceB> ping 10.1.1.1
PING 10.1.1.1: 56 data bytes, press CTRL_C to break
Reply from 10.1.1.1: bytes=56 Sequence=1 ttl=255 time=30 ms
Reply from 10.1.1.1: bytes=56 Sequence=2 ttl=255 time=16 ms
Reply from 10.1.1.1: bytes=56 Sequence=3 ttl=255 time=7 ms
Reply from 10.1.1.1: bytes=56 Sequence=4 ttl=255 time=6 ms
Reply from 10.1.1.1: bytes=56 Sequence=5 ttl=255 time=9 ms

--- 10.1.1.1 ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 6/13/30 ms
<DeviceB> display bgp peer
BGP local router ID : 10.1.1.2
Local AS number : 100
Total number of peers : 1 Peers in established state : 1
```

| Peer     | V | AS  | MsgRcvd | MsgSent | OutQ | Up/Down  | State       | PrefRcv |
|----------|---|-----|---------|---------|------|----------|-------------|---------|
| 10.1.1.1 | 4 | 200 | 5       | 5       | 0    | 00:00:20 | Established | 0       |

- Step 2** Enable the VPN address family in the BGP multi-instance on DeviceB and in the base instance on DeviceC, and establish an EBGP-VPN peer relationship in this address family.

# Configure DeviceB.

```
[~DeviceB] ip vpn-instance vpn1
[*DeviceB-vpn-instance-vpn1] ipv4-family
[*DeviceB-vpn-instance-vpn1-af-ipv4] route-distinguisher 100:3
[*DeviceB-vpn-instance-vpn1-af-ipv4] vpn-target 100:1 export-extcommunity
[*DeviceB-vpn-instance-vpn1-af-ipv4] vpn-target 100:1 import-extcommunity
[*DeviceB-vpn-instance-vpn1-af-ipv4] quit
[*DeviceB-vpn-instance-vpn1] quit
[*DeviceB] bgp 200 instance vpn1
[~DeviceB-bgp-instance-aa] ipv4-family vpn-instance vpn1
[*DeviceB-bgp-instance-aa-vpn1] peer 10.1.2.3 as-number 200
[*DeviceB-bgp-instance-aa-vpn1] quit
[*DeviceB-bgp-instance-aa] quit
[*DeviceB-bgp-instance-aa] interface gigabitethernet2/0/0
[*DeviceB-GigabitEthernet2/0/0] ip binding vpn-instance vpn1
[*DeviceB-GigabitEthernet2/0/0] ip address 10.1.2.2 24
[*DeviceB-GigabitEthernet2/0/0] commit
```

# Configure DeviceC.

```
[~HUAWEI] sysname DeviceC
[*HUAWEI] commit
[~DeviceC] ip vpn-instance vpn1
[*DeviceC-vpn-instance-vpn1] ipv4-family
[*DeviceC-vpn-instance-vpn1-af-ipv4] route-distinguisher 100:3
[*DeviceC-vpn-instance-vpn1-af-ipv4] vpn-target 100:1 export-extcommunity
[*DeviceC-vpn-instance-vpn1-af-ipv4] vpn-target 100:1 import-extcommunity
[*DeviceC-vpn-instance-vpn1-af-ipv4] quit
[*DeviceC-vpn-instance-vpn1] quit
[*DeviceC] bgp 200
[*DeviceC-bgp] ipv4-family vpn-instance vpn1
[*DeviceC-bgp-vpn1] peer 10.1.2.2 as-number 200
[*DeviceC-bgp-vpn1] quit
[*DeviceC-bgp] quit
[*DeviceC-bgp] interface gigabitethernet2/0/0
[*DeviceC-GigabitEthernet2/0/0] ip binding vpn-instance vpn1
[*DeviceC-GigabitEthernet2/0/0] ip address 10.1.2.3 24
[*DeviceC-GigabitEthernet2/0/0] commit
```

After the configuration is complete, check the connectivity between DeviceB and DeviceC. Then, run the **display bgp instance aa vpnv4 all peer** command. The command output shows that the EBGP-VPN peer relationship between DeviceB and DeviceC is **Established**.

DeviceB is used as an example.

```
<DeviceB> ping -vpn-instance vpn1 10.1.2.3
PING 10.1.2.3: 56 data bytes, press CTRL_C to break
Reply from 10.1.2.3: bytes=56 Sequence=1 ttl=255 time=35 ms
Reply from 10.1.2.3: bytes=56 Sequence=2 ttl=255 time=25 ms
Reply from 10.1.2.3: bytes=56 Sequence=3 ttl=255 time=12 ms
Reply from 10.1.2.3: bytes=56 Sequence=4 ttl=255 time=8 ms
Reply from 10.1.2.3: bytes=56 Sequence=5 ttl=255 time=9 ms

--- 10.1.2.3 ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 8/17/35 ms
<DeviceB> display bgp instance aa vpnv4 all peer
BGP local router ID : 10.1.1.2
```

```
Local AS number : 200
Total number of peers : 1
Peers in established state : 1
```

Peer of IPv4-family for vpn instance :

| VPN-Instance | Router ID |     |         |         |      |          |             |         |
|--------------|-----------|-----|---------|---------|------|----------|-------------|---------|
| Peer         | V         | AS  | MsgRcvd | MsgSent | OutQ | Up/Down  | State       | PrefRcv |
| 10.1.2.3     | 4         | 100 | 3       | 3       | 0    | 00:00:03 | Established | 0       |

**Step 3** Configure a static route and import it into the BGP routing table on DeviceA and DeviceC.

# Configure DeviceA.

```
[~DeviceA] ip route-static 192.168.1.1 255.255.255.255 NULL0
[*DeviceA] bgp 100
[*DeviceA-bgp] import-route static
[*DeviceA-bgp] commit
```

# Configure DeviceC.

```
[~DeviceC] ip route-static vpn-instance vpn1 192.168.3.3 255.255.255.255 NULL0
[*DeviceC] bgp 200
[*DeviceC-bgp] ipv4-family vpn-instance vpn1
[*DeviceC-bgp-vpn1] import-route static
[*DeviceC-bgp-vpn1] commit
```

**Step 4** Verify the configuration.

After the preceding configurations are complete, only public network routes can be found on DeviceA.

```
<DeviceA> display bgp routing-table

BGP Local router ID is 10.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network NextHop MED LocPrf PrefVal Path/Ogn
*> 192.168.1.1/32 0.0.0.0 0 0 ?
```

Both public network and VPN routes can be found on DeviceB.

```
<DeviceB> display bgp routing-table

BGP Local router ID is 10.1.1.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 1
Network NextHop MED LocPrf PrefVal Path/Ogn
*> 192.168.1.1/32 10.1.1.1 0 0 ?

<DeviceB> display bgp instance aa vpnv4 all routing-table

BGP Local router ID is 10.1.1.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
```

RPKI validation codes: V - valid, I - invalid, N - not-found

Total number of routes from all PE: 1  
Route Distinguisher: 100:44

| Network           | NextHop  | MED | LocPrf | PrefVal | Path/Ogn |
|-------------------|----------|-----|--------|---------|----------|
| *> 192.168.3.3/32 | 10.1.2.3 | 0   | 0      | 100?    |          |

VPN-Instance vpn1, Router ID 10.1.1.2:

| Network           | NextHop  | MED | LocPrf | PrefVal | Path/Ogn |
|-------------------|----------|-----|--------|---------|----------|
| *> 192.168.3.3/32 | 10.1.2.3 | 0   | 0      | 100?    |          |

Only a VPN route can be found on DeviceC.

<DeviceC> **display bgp vpnv4 all routing-table**

BGP Local router ID is 0.0.0.0

Status codes: \* - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale

Origin : i - IGP, e - EGP, ? - incomplete

RPKI validation codes: V - valid, I - invalid, N - not-found

Total number of routes from all PE: 1  
Route Distinguisher: 100:44

| Network           | NextHop | MED | LocPrf | PrefVal | Path/Ogn |
|-------------------|---------|-----|--------|---------|----------|
| *> 192.168.3.3/32 | 0.0.0.0 | 0   | 0      | ?       |          |

VPN-Instance vpn1, Router ID 10.1.2.3:

| Network           | NextHop | MED | LocPrf | PrefVal | Path/Ogn |
|-------------------|---------|-----|--------|---------|----------|
| *> 192.168.3.3/32 | 0.0.0.0 | 0   | 0      | ?       |          |

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0

bgp 100
peer 10.1.1.2 as-number 100
ipv4-family unicast
undo synchronization
import-route static
peer 10.1.1.2 enable

ip route-static 192.168.1.1 255.255.255.255 NULL0

return
```

- DeviceB configuration file

```

sysname DeviceB

ip vpn-instance vpn1
ipv4-family
route-distinguisher 100:3
apply-label per-instance
vpn-target 100:1 export-extcommunity
vpn-target 100:1 import-extcommunity

interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0

interface GigabitEthernet2/0/0
undo shutdown
ip binding vpn-instance vpn1
ip address 10.1.2.2 255.255.255.0

bgp 100
peer 10.1.1.1 as-number 100
ipv4-family unicast
undo synchronization
peer 10.1.1.1 enable

bgp 200 instance vpn1
ipv4-family vpn-instance vpn1
peer 10.1.2.3 as-number 200

return
```

- DeviceC configuration file

```

sysname DeviceC

ip vpn-instance vpn1
ipv4-family
route-distinguisher 100:3
apply-label per-instance
vpn-target 100:1 export-extcommunity
vpn-target 100:1 import-extcommunity

interface GigabitEthernet2/0/0
undo shutdown
ip binding vpn-instance vpn1
ip address 10.1.2.3 255.255.255.0

bgp 200
ipv4-family unicast
undo synchronization
ipv4-family vpn-instance vpn1
import-route static
peer 10.1.2.2 as-number 200

ip route-static vpn-instance vpn1 192.168.3.3 255.255.255.255 NULL0

return
```

## Example for Configuring a BGP SR LSP

Deploying a complete BGP SR LSP on devices in the same AS helps implement end-to-end service interworking.

## Networking Requirements

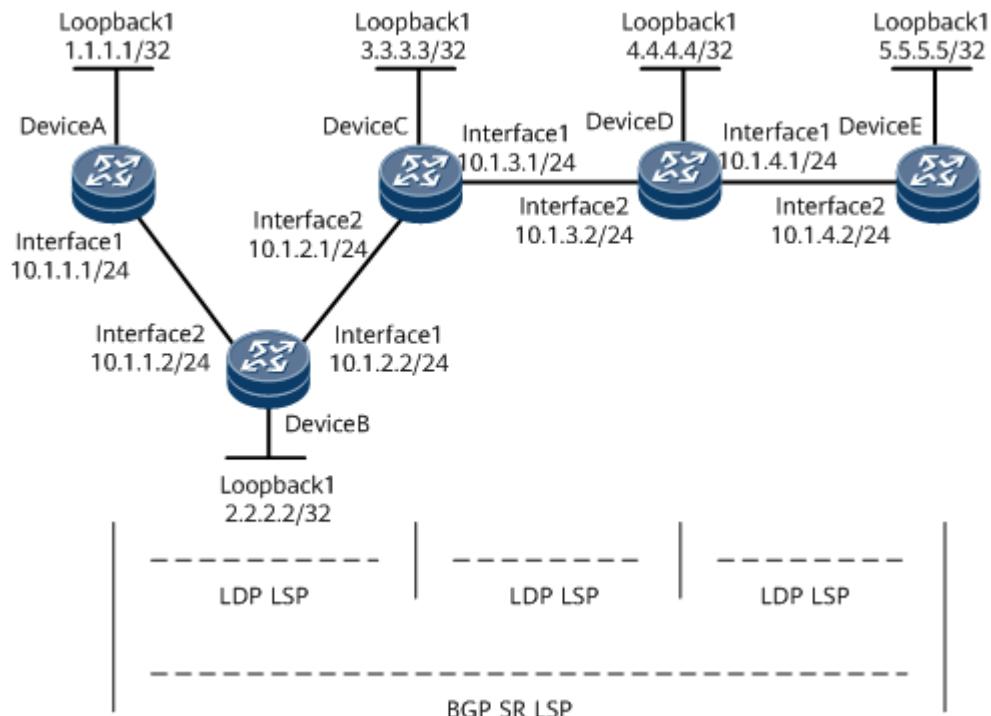
On the network shown in [Figure 1-446](#), OSPF runs between DeviceA and DeviceB, between DeviceB and DeviceC, and between DeviceD and DeviceE. IS-IS runs

between DeviceC and DeviceD. Basic MPLS capabilities and MPLS LDP are configured on DeviceA through DeviceE so that LDP LSPs are established between loopback interfaces of devices in each IGP area. Therefore, traffic between loopback interfaces of devices in each IGP area is encapsulated using MPLS. However, traffic cannot be transmitted across IGP areas because devices cannot ping each other across IGP areas. For example, DeviceA cannot ping DeviceE. To solve this problem, you need to configure an inner MPLS tunnel (BGP SR LSP) from 1.1.1.1 to 5.5.5.5 so that the traffic from 1.1.1.1 to 5.5.5.5 is forwarded through MPLS.

 **NOTE**

In this example, interfaces 1 and 2 represent GE 1/0/0 and GE 2/0/0, respectively.

**Figure 1-446** Configuring a BGP SR LSP



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IP addresses for interfaces.
2. Configure IGP areas.
3. Configure basic MPLS functions and MPLS LDP.

#### 4. Configure a BGP SR LSP.

## Data Preparation

To complete the configuration, you need the following data:

- MPLS LSR IDs of DeviceA through DeviceE
- SRGBs of DeviceA through DeviceE

## Procedure

### Step 1 Configure interface IP addresses.

Take DeviceA as an example.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ip address 10.1.1.1 24
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface loopBack 1
[*DeviceA-LoopBack1] ip address 1.1.1.1 32
[*DeviceA-LoopBack1] commit
[~DeviceA-LoopBack1] quit
```

The interfaces that directly connect the devices can ping each other, but the devices cannot ping each other's loopback interface.

### Step 2 Deploy DeviceA through DeviceC in the same IGP area, and configure OSPF on them to implement interworking in the IGP area.

Take DeviceA as an example.

```
[~DeviceA] ospf 1 router-id 1.1.1.1
[~DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0.0] network 1.1.1.1 0.0.0.0
[*DeviceA-ospf-1-area-0.0.0.0] network 10.1.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0.0] quit
[~DeviceA-ospf-1] quit
```

After the preceding configuration is complete, an OSPF neighbor relationship is established between DeviceA and DeviceB and between DeviceB and DeviceC. If the **display ospf peer** command is run, you can find that the neighbor status is Full. DeviceA and DeviceC can learn each other's Loopback 1 IP address and ping each other successfully.

Deploy DeviceD and DeviceE in another IGP area and configure OSPF on them. Their configurations are similar to the configuration of DeviceA and are not mentioned here.

### Step 3 Deploy DeviceC and DeviceD in the same IGP area, and configure IS-IS on them to implement interworking in the IGP area.

Take DeviceC as an example.

```
[~DeviceC] isis 1
[~DeviceC-isis-1] network-entity 10.0000.0000.0000.0010.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable 1
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface loopBack 1
[*DeviceC-LoopBack1] isis enable 1
```

```
[*DeviceC--LoopBack1] commit
[~DeviceC--LoopBack1] quit
```

After the preceding configuration is complete, DeviceC and DeviceD can learn each other's Loopback 1 IP address and ping each other successfully. However, Loopback 1 interfaces on DeviceA and DeviceD cannot ping each other. This means that OSPF and IS-IS run independently on DeviceC.

**Step 4** Configure basic MPLS functions and MPLS LDP to set up LDP LSPs.

```
Configure basic MPLS functions on DeviceA and enable LDP on the interface connected to DeviceB.
[~DeviceA] mpls lsr-id 1.1.1.1
[*DeviceA] mpls
[*DeviceA-mpls] label advertise non-null
[*DeviceA-mpls] quit
[*DeviceA] mpls ldp
[*DeviceA-mpls-ldp] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] mpls
[*DeviceA-GigabitEthernet1/0/0] mpls ldp
[*DeviceA-GigabitEthernet1/0/0] commit
[~DeviceA-GigabitEthernet1/0/0] quit
```

Configure MPLS on DeviceB through DeviceE. Their configurations are similar to that on DeviceA and are not mentioned here. After the configuration is complete, MPLS forwarding can be implemented between devices in each IGP area.

Check the MPLS label forwarding information base on DeviceA. The command output shows that label 48122 is pushed to the data packet with destination IP address 3.3.3.3/32 and In-Label NULL and that the packet is forwarded through GigabitEthernet1/0/0. Here, DeviceA functions as the ingress.

```
<DeviceA> display mpls lsp include 3.3.3.3 32 verbose
```

```

LSP Information: LDP LSP

No : 1
VrfIndex :
Fec : 3.3.3.3/32
Nexthop : 10.1.1.2
In-Label : NULL
Out-Label : 48122
In-Interface : -----
Out-Interface : GigabitEthernet1/0/0
LspIndex : 5000003
Type : Primary
OutSegmentIndex : 5000002
LsrType : Ingress
Outgoing TunnelType : -----
Outgoing TunnelID : 0x0
Label Operation : PUSH
Mpls-Mtu : 1500
LspAge : 257 sec
Ingress-ELC : Disable

No : 2
VrfIndex :
Fec : 3.3.3.3/32
Nexthop : 10.1.1.2
In-Label : 48124
Out-Label : 48122
In-Interface : -----
Out-Interface : GigabitEthernet2/0/0
LspIndex : 5000003
Type : Primary
OutSegmentIndex : 5000002
```

```
LsrType : Transit
Outgoing TunnelType : -----
Outgoing TunnelID : 0x0
Label Operation : SWAP
Mpls-Mtu : 1500
LspAge : 257 sec
Ingress-ELC : -----
```

Check the MPLS label forwarding information base on DeviceB. After DeviceB receives the packet with In-Label 48122 from DeviceA, DeviceB searches its label forwarding information base, swaps label 48122 with label 48120, and forwards the packet through **GigabitEthernet1/0/0**. Here, DeviceB functions as a transit node.

```
<DeviceB> display mpls lsp include 3.3.3.3 32 verbose
```

```

LSP Information: LDP LSP

No : 1
VrfIndex :
Fec : 3.3.3.3/32
Nexthop : 10.1.2.1
In-Label : NULL
Out-Label: 48120
In-Interface : -----
Out-Interface: GigabitEthernet3/0/1
LspIndex : 5000003
Type : Primary
OutSegmentIndex : 5000002
LsrType : Ingress
Outgoing TunnelType : -----
Outgoing TunnelID : 0x0
Label Operation : PUSH
Mpls-Mtu : 1500
LspAge : 693 sec
Ingress-ELC : Disable

No : 2
VrfIndex :
Fec : 3.3.3.3/32
Nexthop : 10.1.2.1
In-Label : 48122
Out-Label: 48120
In-Interface : -----
Out-Interface: GigabitEthernet1/0/0
LspIndex : 5000003
Type : Primary
OutSegmentIndex : 5000002
LsrType : Transit
Outgoing TunnelType : -----
Outgoing TunnelID : 0x0
Label Operation : SWAP
Mpls-Mtu : 1500
LspAge : 693 sec
Ingress-ELC : -----
```

Check the MPLS label forwarding information base on DeviceC. After DeviceC receives the packet carrying In-Label 48120 from DeviceB, DeviceC searches its label forwarding information base, and pops out the label. Then the packet reaches its destination. Here, DeviceC functions as the egress.

```
<DeviceC> display mpls lsp include 3.3.3.3 32 verbose
```

```

LSP Information: LDP LSP

No : 1
VrfIndex :
Fec : 3.3.3.3/32
```

```
Nexthop : 127.0.0.1
In-Label : 48120
Out-Label : NULL
In-Interface : -----
Out-Interface: -----
LspIndex : 5000001
Type : Primary
OutSegmentIndex: 4294967295
LsrType : Egress
Outgoing TunnelType: -----
Outgoing TunnelID : 0x0
Label Operation : POP
Mpls-Mtu : -----
LspAge : 1212 sec
Ingress-ELC : -----
```

After the preceding configuration is complete, an LDP LSP is established in the IGP area, and data packets can be forwarded using MPLS labels only in the IGP area. When a packet reaches the IGP area border, its label is popped out, and the label-based forwarding process is complete, indicating that the packet cannot be further forwarded. Therefore, an inner MPLS tunnel (inner BGP SR LSP) needs to be established from 1.1.1.1 to 5.5.5.5 so that each MPLS packet carries two MPLS labels. When a packet reaches the IGP area border, the outer label is popped out, and there is still an inner MPLS label in the packet, according to which devices can forward the packet across IGP areas, ensuring whole-process MPLS forwarding and achieving end-to-end service interworking.

**Step 5** Configure a BGP SR LSP.

a. Configure SRGBs.

```
Configure DeviceA.
```

```
[~DeviceA] bgp 100
[*DeviceA-bgp] segment-routing global-block 16000 17000
[*DeviceA-bgp] quit
[*DeviceA] commit
```

```
Configure DeviceC.
```

```
[~DeviceC] bgp 100
[*DeviceC-bgp] segment-routing global-block 18000 19000
[*DeviceC-bgp] quit
[*DeviceC] commit
```

Configure the SRGB ranges [20000-21000] and [22000-23000] for DeviceD and DeviceE, respectively. The configurations on DeviceD and DeviceE are similar to the configuration on DeviceC and are not mentioned here.

b. Configure BGP peer relationships.

```
Configure DeviceA.
```

```
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 3.3.3.3 as-number 100
[*DeviceA-bgp] peer 3.3.3.3 connect-interface LoopBack1
[*DeviceA-bgp] quit
[*DeviceA] commit
```

```
Configure DeviceC.
```

```
[~DeviceC] bgp 100
[*DeviceC-bgp] peer 1.1.1.1 as-number 100
[*DeviceC-bgp] peer 1.1.1.1 connect-interface LoopBack1
[*DeviceC-bgp] peer 4.4.4.4 as-number 100
[*DeviceC-bgp] peer 4.4.4.4 connect-interface LoopBack1
```

```
[*DeviceC-bgp] quit
[*DeviceC] commit
```

The configurations on DeviceD and DeviceE are similar to the configuration on DeviceC and are not mentioned here.

c. Configure DeviceC and DeviceD as route reflectors (RRs) so that DeviceA and DeviceE can learn BGP routes from each other.

# Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] peer 1.1.1.1 reflect-client
[*DeviceC-bgp] peer 1.1.1.1 next-hop-local
[*DeviceC-bgp] peer 4.4.4.4 reflect-client
[*DeviceC-bgp] peer 4.4.4.4 next-hop-local
[*DeviceC-bgp] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] bgp 100
[*DeviceD-bgp] peer 3.3.3.3 reflect-client
[*DeviceD-bgp] peer 3.3.3.3 next-hop-local
[*DeviceD-bgp] peer 5.5.5.5 reflect-client
[*DeviceD-bgp] peer 5.5.5.5 next-hop-local
[*DeviceD-bgp] quit
[*DeviceD] commit
```

d. Configure BGP SR LSP ingresses.

# Configure DeviceA.

```
[~DeviceA] route-policy policy1 permit node 1
[*DeviceA-route-policy] apply mpls-label
[*DeviceA-route-policy] quit
[*DeviceA] bgp 100
[*DeviceA-bgp] network 1.1.1.1 255.255.255.255 label-index 10
[*DeviceA-bgp] peer 3.3.3.3 route-policy policy1 export
[*DeviceA-bgp] peer 3.3.3.3 label-route-capability
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 3.3.3.3 prefix-sid
[*DeviceA-bgp-af-ipv4] quit
[*DeviceA-bgp] quit
[*DeviceA] commit
```

# Configure DeviceE.

```
[~DeviceE] route-policy policy1 permit node 1
[*DeviceE-route-policy] apply mpls-label
[*DeviceE-route-policy] quit
[*DeviceE] bgp 100
[*DeviceE-bgp] network 5.5.5.5 255.255.255.255 label-index 50
[*DeviceE-bgp] peer 4.4.4.4 route-policy policy1 export
[*DeviceE-bgp] peer 4.4.4.4 label-route-capability
[*DeviceE-bgp] ipv4-family unicast
[*DeviceE-bgp-af-ipv4] peer 4.4.4.4 prefix-sid
[*DeviceE-bgp-af-ipv4] quit
[*DeviceE-bgp] quit
[*DeviceE] commit
```

e. Configure transit nodes for the BGP SR LSP.

# Configure DeviceC.

```
[~DeviceC] route-policy policy1 permit node 1
[*DeviceC-route-policy] if-match mpls-label
[*DeviceC-route-policy] apply mpls-label
[*DeviceC-route-policy] quit
```

```
[*DeviceC] bgp 100
[*DeviceC-bgp] peer 1.1.1.1 label-route-capability
[*DeviceC-bgp] peer 4.4.4.4 label-route-capability
[*DeviceC-bgp] peer 1.1.1.1 route-policy policy1 export
[*DeviceC-bgp] peer 4.4.4.4 route-policy policy1 export
[*DeviceC-bgp] ipv4-family unicast
[*DeviceC-bgp-af-ipv4] peer 1.1.1.1 prefix-sid
[*DeviceC-bgp-af-ipv4] peer 4.4.4.4 prefix-sid
[*DeviceC-bgp-af-ipv4] quit
[*DeviceC-bgp] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] route-policy policy1 permit node 1
[*DeviceD-route-policy] if-match mpls-label
[*DeviceD-route-policy] apply mpls-label
[*DeviceD-route-policy] quit
[*DeviceD] bgp 100
[*DeviceD-bgp] peer 3.3.3.3 label-route-capability
[*DeviceD-bgp] peer 5.5.5.5 label-route-capability
[*DeviceD-bgp] peer 3.3.3.3 route-policy policy1 export
[*DeviceD-bgp] peer 5.5.5.5 route-policy policy1 export
[*DeviceD-bgp] ipv4-family unicast
[*DeviceD-bgp-af-ipv4] peer 3.3.3.3 prefix-sid
[*DeviceD-bgp-af-ipv4] peer 5.5.5.5 prefix-sid
[*DeviceD-bgp-af-ipv4] quit
[*DeviceD-bgp] quit
[*DeviceD] commit
```

After the configuration is complete, run the **display mpls lsp** command on DeviceE to view information about BGP LSPs. The command output shows that the In Label of the route with the destination IP address 5.5.5.5/32 is 22050. In this case, DeviceE instructs DeviceD to use 22050 as the BGP SR LSP label for the route with the destination IP address 5.5.5.5/32. DeviceE then creates the entry in its BGP LSP table.

```
[~DeviceE] display mpls lsp
Flag after Out IF: (I) - RLFA Iterated LSP, (I*) - Normal and RLFA Iterated LSP
Flag after LDP FRR: (L) - Logic FRR LSP
```

| LSP Information: LDP LSP |              |            |          |
|--------------------------|--------------|------------|----------|
| FEC                      | In/Out Label | In/Out IF  | Vrf Name |
| 4.4.4.4/32               | NULL/48120   | -/Eth3/0/3 |          |
| 4.4.4.4/32               | 48121/48120  | -/Eth3/0/3 |          |
| 5.5.5.5/32               | 48120/NULL   | -/-        |          |

| LSP Information: BGP LSP |              |           |          |
|--------------------------|--------------|-----------|----------|
| FEC                      | In/Out Label | In/Out IF | Vrf Name |
| 1.1.1.1/32               | NULL/20010   | -/-       |          |
| 5.5.5.5/32               | 22050/NULL   | -/-       |          |

Run the **display mpls lsp** command on DeviceC to view information about BGP LSPs. The command output shows that the In Label of the route with the destination IP address 5.5.5.5/32 is 18050. This label is notified to DeviceC by DeviceD, and DeviceC is instructed to use 18050 as the BGP SR LSP label for the route with the destination IP address 5.5.5.5/32. DeviceC then creates the entry in its BGP LSP table. In addition, the Out Label is 20050, which is notified by DeviceD.

```
[~DeviceC] display mpls lsp
Flag after Out IF: (I) - RLFA Iterated LSP, (I*) - Normal and RLFA Iterated LSP
Flag after LDP FRR: (L) - Logic FRR LSP
```

| LSP Information: LDP LSP |              |            |          |
|--------------------------|--------------|------------|----------|
| FEC                      | In/Out Label | In/Out IF  | Vrf Name |
| 4.4.4.4/32               | NULL/48120   | -/Eth3/0/3 |          |
| 4.4.4.4/32               | 48121/48120  | -/Eth3/0/3 |          |

| FEC        | In/Out Label | In/Out IF  | Vrf Name |
|------------|--------------|------------|----------|
| 1.1.1.1/32 | NULL/48121   | -/Eth3/0/1 |          |
| 1.1.1.1/32 | 48121/48121  | -/Eth3/0/1 |          |
| 2.2.2.2/32 | NULL/48120   | -/Eth3/0/1 |          |
| 2.2.2.2/32 | 48122/48120  | -/Eth3/0/1 |          |
| 3.3.3.3/32 | 48120/NULL   | -/-        |          |
| 4.4.4.4/32 | NULL/48120   | -/Eth3/0/2 |          |
| 4.4.4.4/32 | 48123/48120  | -/Eth3/0/2 |          |

LSP Information: BGP LSP

| FEC        | In/Out Label       | In/Out IF | Vrf Name |
|------------|--------------------|-----------|----------|
| 1.1.1.1/32 | 18010/16010        | -/-       |          |
| 5.5.5.5/32 | <b>18050/20050</b> | -/-       |          |
| 5.5.5.5/32 | NULL/20050         | -/-       |          |

Now, two MPLS labels are available, one for the LDP LSP, and the other for the BGP LSP. Then, MPLS forwarding from DeviceA to DeviceE is implemented.

**Step 6** Verify the configuration.

After the preceding configuration is complete, DeviceA and DeviceE can learn the routes to each other's interfaces and ping each other's Loopback1 interface.

Take the command output on DeviceA as an example.

```
<DeviceA> ping -a 1.1.1.1 5.5.5.5
PING 5.5.5.5: 56 data bytes, press CTRL_C to break
 Reply from 5.5.5.5: bytes=56 Sequence=1 ttl=252 time=30 ms
 Reply from 5.5.5.5: bytes=56 Sequence=2 ttl=252 time=23 ms
 Reply from 5.5.5.5: bytes=56 Sequence=3 ttl=252 time=26 ms
 Reply from 5.5.5.5: bytes=56 Sequence=4 ttl=252 time=28 ms
 Reply from 5.5.5.5: bytes=56 Sequence=5 ttl=252 time=22 ms

--- 5.5.5.5 ping statistics ---
 5 packet(s) transmitted
 5 packet(s) received
 0.00% packet loss
 round-trip min/avg/max = 22/25/30 ms
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA
#
mpls lsr-id 1.1.1.1
#
mpls
label advertise non-null
#
mpls ldp
#
ipv4-family
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 1.1.1.1 255.255.255.255
#
bgp 100
```

```
segment-routing global-block 16000 17000
peer 3.3.3.3 as-number 100
peer 3.3.3.3 connect-interface LoopBack1
#
ipv4-family unicast
undo synchronization
network 1.1.1.1 255.255.255.255 label-index 10
peer 3.3.3.3 enable
peer 3.3.3.3 route-policy policy1 export
peer 3.3.3.3 label-route-capability
peer 3.3.3.3 prefix-sid
#
ospf 1 router-id 1.1.1.1
area 0.0.0.0
network 1.1.1.1 0.0.0.0
network 10.1.1.0 0.0.0.255
#
route-policy policy1 permit node 1
apply mpls-label
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
mpls lsr-id 2.2.2.2
#
mpls
label advertise non-null
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.0
mpls
mpls ldp
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 2.2.2.2 255.255.255.255
#
ospf 1 router-id 2.2.2.2
area 0.0.0.0
network 2.2.2.2 0.0.0.0
network 10.1.1.0 0.0.0.255
network 10.1.2.0 0.0.0.255
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
mpls lsr-id 3.3.3.3
#
mpls
label advertise non-null
#
mpls ldp
#
isis 1
network-entity 10.0000.0000.0000.0010.00
#
```

```
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.0
isis enable 1
mpls
mpls ldp
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 3.3.3.3 255.255.255.255
isis enable 1
#
bgp 100
segment-routing global-block 18000 19000
peer 1.1.1.1 as-number 100
peer 1.1.1.1 connect-interface LoopBack1
peer 4.4.4.4 as-number 100
peer 4.4.4.4 connect-interface LoopBack1
#
ipv4-family unicast
undo synchronization
peer 1.1.1.1 enable
peer 1.1.1.1 route-policy policy1 export
peer 1.1.1.1 reflect-client
peer 1.1.1.1 next-hop-local
peer 1.1.1.1 label-route-capability
peer 1.1.1.1 prefix-sid
peer 4.4.4.4 enable
peer 4.4.4.4 route-policy policy1 export
peer 4.4.4.4 reflect-client
peer 4.4.4.4 next-hop-local
peer 4.4.4.4 label-route-capability
peer 4.4.4.4 prefix-sid
#
ospf 1 router-id 3.3.3.3
area 0.0.0
network 3.3.3.3 0.0.0.0
network 10.1.2.0 0.0.0.255
#
route-policy policy1 permit node 1
if-match mpls-label
apply mpls-label
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
mpls lsr-id 4.4.4.4
#
mpls
label advertise non-null
#
mpls ldp
#
isis 1
network-entity 10.0000.0000.0000.0020.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.4.1 255.255.255.0
mpls
mpls ldp
#
```

```
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.3.2 255.255.255.0
isis enable 1
mpls
mpls ldp
#
interface LoopBack1
ip address 4.4.4.4 255.255.255.255
isis enable 1
#
bgp 100
segment-routing global-block 20000 21000
peer 3.3.3.3 as-number 100
peer 3.3.3.3 connect-interface LoopBack1
peer 5.5.5.5 as-number 100
peer 5.5.5.5 connect-interface LoopBack1
#
ipv4-family unicast
undo synchronization
peer 3.3.3.3 enable
peer 3.3.3.3 route-policy policy1 export
peer 3.3.3.3 reflect-client
peer 3.3.3.3 next-hop-local
peer 3.3.3.3 label-route-capability
peer 3.3.3.3 prefix-sid
peer 5.5.5.5 enable
peer 5.5.5.5 route-policy policy1 export
peer 5.5.5.5 reflect-client
peer 5.5.5.5 next-hop-local
peer 5.5.5.5 label-route-capability
peer 5.5.5.5 prefix-sid
#
ospf 2 router-id 4.4.4.4
area 0.0.0
network 4.4.4.4 0.0.0.0
network 10.1.4.0 0.0.0.255
#
route-policy policy1 permit node 1
if-match mpls-label
apply mpls-label
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
mpls lsr-id 5.5.5.5
#
mpls
label advertise non-null
#
mpls ldp
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 10.1.4.2 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 5.5.5.5 255.255.255.255
#
bgp 100
segment-routing global-block 22000 23000
peer 4.4.4.4 as-number 100
peer 4.4.4.4 connect-interface LoopBack1
#
ipv4-family unicast
```

```
undo synchronization
network 5.5.5.5 255.255.255.255 label-index 50
peer 4.4.4.4 enable
peer 4.4.4.4 route-policy policy1 export
peer 4.4.4.4 label-route-capability
peer 4.4.4.4 prefix-sid
#
ospf 2 router-id 5.5.5.5
area 0.0.0.0
network 5.5.5.5 0.0.0.0
network 10.1.4.0 0.0.0.255
#
route-policy policy1 permit node 1
apply mpls-label
#
return
```

## Example for Configuring a BGP Virtual Link

An SRv6 EPE virtual link can be created for a BGP multi-hop peer relationship to implement communication across a third-party network.

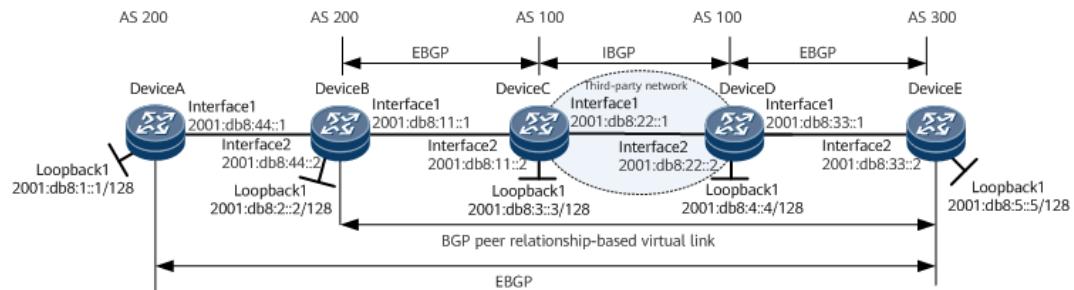
## Networking Requirements

In [Figure 1-447](#), a third-party network is deployed between DeviceC and DeviceD. The two devices run IS-IS for basic routing and establish an IBGP peer relationship. IPv6 EBGP peer relationships are established between directly connected interfaces of DeviceB and DeviceC, and between directly connected interfaces of DeviceD and DeviceE. To establish an SRv6 TE Policy across the third-party network, you can configure a BGP peer relationship-based virtual link between DeviceB and DeviceE. In addition, attribute information required for SRv6 TE Policy path computation is provided, such as the link latency, metric, affinity, and SRLG. This prevents the mismatch of the address information in the link information reported by DeviceB and DeviceE through BGP-LS, which would otherwise cause a link combination failure on the controller.

### NOTE

Interface1 and interface2 in this example represent GE1/0/0 and GE2/0/0, respectively.

**Figure 1-447** Configuring a BGP virtual link



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Assign an IPv6 address to each interface on DeviceA through DeviceE.
2. Enable IS-IS, configure an IS-IS level, and specify a network entity title (NET) for DeviceC and DeviceD.
3. Configure a VPN instance on DeviceC and DeviceD.
4. Establish an IBGP peer relationship between DeviceC and DeviceD.
5. Establish an EBGP peer relationship between DeviceB and DeviceC and another one between DeviceD and DeviceE.
6. Configure SRv6 SIDs and IS-IS SRv6 and configure VPN routes to carry the SID attribute on DeviceC and DeviceD.
7. Configure an SRv6 TE Policy on DeviceC and DeviceD.
8. Configure a tunnel policy on DeviceC and DeviceD to import VPN traffic.
9. Establish an EBGP peer relationship between DeviceB and DeviceE.
10. Configure a VPN instance on DeviceA and DeviceE.
11. Establish an EBGP peer relationship between DeviceA and DeviceE.
12. Configure SRv6 SIDs and enable IS-IS SRv6 on DeviceA, DeviceB, and DeviceE. Configure VPN routes to carry the SID attribute on DeviceA and DeviceE.
13. Configure the BGP EPE and virtual link functions on DeviceB and DeviceE.
14. Configure an SRv6 TE Policy on DeviceA and DeviceE.
15. Configure a tunnel policy on DeviceA and DeviceE to import VPN traffic.

## Data Preparation

To complete the configuration, you need the following data:

- IPv6 addresses of interfaces on DeviceA through DeviceE
- IS-IS process IDs and levels on DeviceA through DeviceD
- VPN instance names, RDs, and RTs on DeviceA, DeviceE, DeviceC, and DeviceD

## Procedure

### Step 1 Configure IPv6 addresses and enable IPv6 forwarding for interfaces.

# Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 1/0/0
[~DeviceA-GigabitEthernet1/0/0] ipv6 enable
[*DeviceA-GigabitEthernet1/0/0] ipv6 address 2001:db8:44::1/96
[*DeviceA-GigabitEthernet1/0/0] quit
[~DeviceA] interface loopBack 1
[~DeviceA-LoopBack1] ipv6 enable
[*DeviceA-LoopBack1] ipv6 address 2001:db8:1::1/128
[*DeviceA-LoopBack1] commit
[~DeviceA-LoopBack1] quit
```

# Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] interface gigabitethernet 1/0/0
```

```
[~DeviceB-GigabitEthernet1/0/0] ipv6 enable
[*DeviceB-GigabitEthernet1/0/0] ipv6 address 2001:db8:11::1/96
[*DeviceB-GigabitEthernet1/0/0] quit
[~DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] ipv6 enable
[*DeviceB-GigabitEthernet2/0/0] ipv6 address 2001:db8:44::2/96
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface loopBack 1
[*DeviceB-LoopBack1] ipv6 enable
[*DeviceB-LoopBack1] ipv6 address 2001:db8:2::2/128
[*DeviceB-LoopBack1] commit
[~DeviceB-LoopBack1] quit
```

# Configure DeviceC.

```
<DeviceC> system-view
[~DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] ipv6 enable
[*DeviceC-GigabitEthernet1/0/0] ipv6 address 2001:db8:22::1/96
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ipv6 enable
[*DeviceC-GigabitEthernet2/0/0] ipv6 address 2001:db8:11::2/96
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] interface loopBack 1
[*DeviceC-LoopBack1] ipv6 enable
[*DeviceC-LoopBack1] ipv6 address 2001:db8:3::3/128
[*DeviceC-LoopBack1] commit
[~DeviceC-LoopBack1] quit
```

# Configure DeviceD.

```
<DeviceD> system-view
[~DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] ipv6 enable
[*DeviceD-GigabitEthernet1/0/0] ipv6 address 2001:db8:33::1/96
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] ipv6 enable
[*DeviceD-GigabitEthernet2/0/0] ipv6 address 2001:db8:22::2/96
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] interface loopBack 1
[*DeviceD-LoopBack1] ipv6 enable
[*DeviceD-LoopBack1] ipv6 address 2001:db8:4::4/128
[*DeviceD-LoopBack1] commit
[~DeviceD-LoopBack1] quit
```

# Configure DeviceE.

```
<DeviceE> system-view
[~DeviceE] interface gigabitethernet 2/0/0
[*DeviceE-GigabitEthernet2/0/0] ipv6 enable
[*DeviceE-GigabitEthernet2/0/0] ipv6 address 2001:db8:33::2/96
[*DeviceE-GigabitEthernet2/0/0] quit
[*DeviceE] interface loopBack 1
[*DeviceE-LoopBack1] ipv6 enable
[*DeviceE-LoopBack1] ipv6 address 2001:db8:5::5/128
[*DeviceE-LoopBack1] commit
[~DeviceE-LoopBack1] quit
```

The interfaces that directly connect the devices can ping each other, but the devices cannot ping each other's loopback interface.

## Step 2 Configure IS-IS.

# Configure DeviceA.

```
[~DeviceA] isis 1
[*DeviceA-isis-1] is-level level-1
```

```
[*DeviceA-isis-1] cost-style wide
[*DeviceA-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceA-isis-1] ipv6 enable topology ipv6
[*DeviceA-isis-1] quit
[*DeviceA] interface gigabitethernet 1/0/0
[*DeviceA-GigabitEthernet1/0/0] isis ipv6 enable 1
[*DeviceA-GigabitEthernet1/0/0] quit
[*DeviceA] interface loopBack 1
[*DeviceA-LoopBack1] isis ipv6 enable 1
[*DeviceA-LoopBack1] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] is-level level-1
[*DeviceB-isis-1] cost-style wide
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] ipv6 enable topology ipv6
[*DeviceB-isis-1] ipv6 import-route bgp level-1
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] isis ipv6 enable 1
[*DeviceB-GigabitEthernet2/0/0] quit
[*DeviceB] interface loopBack 1
[*DeviceB-LoopBack1] isis ipv6 enable 1
[*DeviceB-LoopBack1] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
[~DeviceC] isis 100
[*DeviceC-isis-100] is-level level-2
[*DeviceC-isis-100] cost-style wide
[*DeviceC-isis-100] network-entity 20.0000.0000.0001.00
[*DeviceC-isis-100] ipv6 enable topology ipv6
[*DeviceC-isis-100] quit
[*DeviceC] interface gigabitethernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis ipv6 enable 100
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface loopBack 1
[*DeviceC-LoopBack1] isis ipv6 enable 100
[*DeviceC-LoopBack1] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] isis 100
[*DeviceD-isis-1] is-level level-2
[*DeviceD-isis-1] cost-style wide
[*DeviceD-isis-1] network-entity 20.0000.0000.0002.00
[*DeviceD-isis-1] ipv6 enable topology ipv6
[*DeviceD-isis-1] quit
[*DeviceD] interface gigabitethernet 2/0/0
[*DeviceD-GigabitEthernet2/0/0] isis ipv6 enable 100
[*DeviceD-GigabitEthernet2/0/0] quit
[*DeviceD] interface loopBack 1
[*DeviceD-LoopBack1] isis ipv6 enable 100
[*DeviceD-LoopBack1] quit
[*DeviceD] commit
```

After the configuration is complete, perform the following operations to check whether IS-IS is successfully configured.

# Display IS-IS neighbor information. DeviceC is used as an example.

```
[~DeviceC] display isis peer
```

```
Peer information for ISIS(100)
```

| System Id       | Interface | Circuit Id        | State | HoldTime | Type | PRI |
|-----------------|-----------|-------------------|-------|----------|------|-----|
| 0000.0000.0002* | GE1/0/0   | 0000.0000.0002.02 | Up    | 28s      | L2   | 64  |

Total Peer(s): 1

# Display IS-IS routing table information. DeviceC is used as an example.

[~DeviceC] **display isis route**

ISIS(100) Level-2 Forwarding Table

| IPV6 Dest.                    | ExitInterface             | NextHop | Cost   | Flags |
|-------------------------------|---------------------------|---------|--------|-------|
| 2001:db8:3::3/1 Loop1<br>28   | Direct                    | 0       | D/-L/- |       |
| 2001:db8:4::4/1 GE1/0/0<br>28 | FE80::865B:12FF:FE75:E73D | 10      | A/-/-  |       |
| 2001:db8:11::/9 GE2/0/0<br>6  | Direct                    | 10      | D/-L/- |       |
| 2001:db8:22::/9 GE1/0/0<br>6  | Direct                    | 10      | D/-L/- |       |

Flags: D-Direct, A-Added to URT, L-Advertised in LSPs, S-IGP Shortcut,  
U-Up/Down Bit Set, LP-Local Prefix-Sid  
Protect Type: L-Link Protect, N-Node Protect

- Step 3** Create a VPN instance and configure the IPv6 address family capability for it on DeviceC and DeviceD, and implement access of DeviceB to DeviceC and access of DeviceE to DeviceD.

# Configure DeviceC.

```
[~DeviceC] ip vpn-instance vrftest
[*DeviceC-vpn-instance-vrftest] ipv6-family
[*DeviceC-vpn-instance-vrftest-af-ipv6] route-distinguisher 1:1
[*DeviceC-vpn-instance-vrftest-af-ipv6] vpn-target 100:1 export-extcommunity
[*DeviceC-vpn-instance-vrftest-af-ipv6] vpn-target 100:1 import-extcommunity
[*DeviceC-vpn-instance-vrftest-af-ipv6] quit
[*DeviceC-vpn-instance-vrftest] quit
[*DeviceC] interface gigabitethernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] ip binding vpn-instance vrftest
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] ip vpn-instance vrftest
[*DeviceD-vpn-instance-vrftest] ipv6-family
[*DeviceD-vpn-instance-vrftest-af-ipv6] route-distinguisher 1:1
[*DeviceD-vpn-instance-vrftest-af-ipv6] vpn-target 100:1 export-extcommunity
[*DeviceD-vpn-instance-vrftest-af-ipv6] vpn-target 100:1 import-extcommunity
[*DeviceD-vpn-instance-vrftest-af-ipv6] quit
[*DeviceD-vpn-instance-vrftest] quit
[*DeviceD] interface gigabitethernet 1/0/0
[*DeviceD-GigabitEthernet1/0/0] ip binding vpn-instance vrftest
[*DeviceD-GigabitEthernet1/0/0] quit
[*DeviceD] commit
```

- Step 4** Establish an IBGP peer relationship between DeviceC and DeviceD.

# Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] router-id 2.22.2.2
[*DeviceC-bgp] peer 2001:db8:4::4 as-number 100
[*DeviceC-bgp] peer 2001:db8:4::4 connect-interface LoopBack1
[*DeviceC-bgp] ipv6-family vpngv6
[*DeviceC-bgp-af-vpngv6] peer 2001:db8:4::4 enable
```

```
[*DeviceC-bgp-af-vpnv6] quit
[*DeviceC-bgp] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] bgp 100
[*DeviceD-bgp] router-id 3.33.3.3
[*DeviceD-bgp] peer 2001:db8:3::3 as-number 100
[*DeviceD-bgp] peer 2001:db8:3::3 connect-interface LoopBack1
[*DeviceD-bgp] ipv6-family vpnv6
[*DeviceD-bgp-af-vpnv6] peer 2001:db8:3::3 enable
[*DeviceD-bgp-af-vpnv6] quit
[*DeviceD-bgp] quit
[*DeviceD] commit
```

After the configuration is complete, run the **display bgp vpnv6 all peer** command on DeviceC. The command output shows that the IBGP peer relationship between DeviceC and DeviceD is in Established state.

```
[~DeviceC] display bgp vpnv6 all peer

BGP local router ID : 2.22.2.2
Local AS number : 100
Total number of peers : 1 Peers in established state : 1

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
2001:db8:4::4 4 100 37 40 0 00:29:43 Established 0
```

**Step 5** Establish an EBGP peer relationship between DeviceB and DeviceC and another one between DeviceD and DeviceE.

# Configure DeviceB.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 1.11.1.1
[*DeviceB-bgp] peer 2001:db8:11::2 as-number 100
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:11::2 enable
[*DeviceB-bgp-af-ipv6] network 2001:db8:2::2 128
[*DeviceB-bgp-af-ipv6] quit
[*DeviceB-bgp] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] ipv6-family vpn-instance vrftest
[*DeviceC-bgp-6-vrftest] peer 2001:db8:11::1 as-number 200
[*DeviceC-bgp-6-vrftest] quit
[*DeviceC-bgp] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] bgp 100
[*DeviceD-bgp] ipv6-family vpn-instance vrftest
[*DeviceD-bgp-6-vrftest] peer 2001:db8:33::2 as-number 300
[*DeviceD-bgp-6-vrftest] quit
[*DeviceD-bgp] quit
[*DeviceD] commit
```

# Configure DeviceE.

```
[~DeviceE] bgp 300
[*DeviceE-bgp] router-id 4.44.4.4
[*DeviceE-bgp] peer 2001:db8:33::1 as-number 100
[*DeviceE-bgp] ipv6-family unicast
[*DeviceE-bgp-af-ipv6] peer 2001:db8:33::1 enable
```

```
[*DeviceE-bgp-af-ipv6] network 2001:db8:5::5 128
[*DeviceE-bgp-af-ipv6] quit
[*DeviceE-bgp] quit
[*DeviceE] commit
```

After the configuration is complete, run the **display bgp vpnv6 vpn-instance peer** command to check whether a BGP peer relationship has been established between DeviceB and DeviceC, and between DeviceD and DeviceE. If the **Established** state is displayed in the command output, the BGP peer relationship has been established successfully.

DeviceC is used as an example.

```
[~DeviceC] display bgp vpnv6 vpn-instance vrftest peer

BGP local router ID : 2.22.2.2
Local AS number : 100
Total number of peers : 1 Peers in established state : 1

VPN-Instance vrftest, Router ID 2.22.2.2:
Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
2001:db8:11::1 4 200 7 5 0 00:02:29 Established 1
```

**Step 6** Configure SRv6 SIDs and configure VPN routes to carry the SID attribute on DeviceC and DeviceD.

# Configure DeviceC.

```
[~DeviceC] segment-routing ipv6
[*DeviceC-segment-routing-ipv6] encapsulation source-address 2001:db8:3::3
[*DeviceC-segment-routing-ipv6] locator locator1 ipv6-prefix 2001:db8:333:: 64 static 32
[*DeviceC-segment-routing-ipv6-locator] opcode 2001:db8:333::333 end psp
[*DeviceC-segment-routing-ipv6-locator] quit
[*DeviceC-segment-routing-ipv6] quit
[*DeviceC] bgp 100
[*DeviceC-bgp] ipv6-family vpnv6
[*DeviceC-bgp-af-vpnv6] peer 2001:db8:4::4 prefix-sid
[*DeviceC-bgp-af-vpnv6] quit
[*DeviceC-bgp] ipv6-family vpn-instance vrftest
[*DeviceC-bgp-6-vrftest] segment-routing ipv6 traffic-engineer best-effort
[*DeviceC-bgp-6-vrftest] segment-routing ipv6 locator locator1
[*DeviceC-bgp-6-vrftest] quit
[*DeviceC-bgp] quit
[*DeviceC] isis 100
[*DeviceC-isis-100] segment-routing ipv6 locator locator1 auto-sid-disable
[*DeviceC-isis-100] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] segment-routing ipv6
[*DeviceD-segment-routing-ipv6] encapsulation source-address 2001:db8:4::4
[*DeviceD-segment-routing-ipv6] locator locator1 ipv6-prefix 2001:db8:444:: 64 static 32
[*DeviceD-segment-routing-ipv6-locator] opcode 2001:db8:444::444 end psp
[*DeviceD-segment-routing-ipv6-locator] quit
[*DeviceD-segment-routing-ipv6] quit
[*DeviceD] bgp 100
[*DeviceD-bgp] ipv6-family vpnv6
[*DeviceD-bgp-af-vpnv6] peer 2001:db8:3::3 prefix-sid
[*DeviceD-bgp-af-vpnv6] quit
[*DeviceD-bgp] ipv6-family vpn-instance vrftest
[*DeviceD-bgp-6-vrftest] segment-routing ipv6 traffic-engineer best-effort
[*DeviceD-bgp-6-vrftest] segment-routing ipv6 locator locator1
[*DeviceD-bgp-6-vrftest] quit
[*DeviceD-bgp] quit
[*DeviceD] isis 100
[*DeviceD-isis-100] segment-routing ipv6 locator locator1 auto-sid-disable
[*DeviceD-isis-100] quit
[*DeviceD] commit
```

Run the **display segment-routing ipv6 local-sid end forwarding** command to check information about the SRv6 local SID table.

```
[~DeviceC] display segment-routing ipv6 local-sid end forwarding
```

My Local-SID End Forwarding Table

```

SID : 2001:db8:333::333/128 FuncType : End
Flavor : PSP
LocatorName : locator1 LocatorID : 2
ProtocolType: STATIC ProcessID : --
UpdateTime : 2022-02-11 17:06:25.197
```

Total SID(s): 1

```
[~DeviceD] display segment-routing ipv6 local-sid end forwarding
```

My Local-SID End Forwarding Table

```

SID : 2001:db8:444::444/128 FuncType : End
Flavor : PSP
LocatorName : locator1 LocatorID : 1
ProtocolType: STATIC ProcessID : --
UpdateTime : 2022-02-11 17:08:23.955
```

Total SID(s): 1

### Step 7 Configure an SRv6 TE Policy.

# Configure DeviceC.

```
[~DeviceC] segment-routing ipv6
[*DeviceC-segment-routing-ipv6] segment-list s1
[*DeviceC-segment-routing-ipv6-segment-list-s1] index 5 sid ipv6 2001:db8:444::444
[*DeviceC-segment-routing-ipv6-segment-list-s1] quit
[*DeviceC-segment-routing-ipv6] srv6-te-policy locator locator1
[*DeviceC-segment-routing-ipv6] srv6-te policy policy1 endpoint 2001:db8:4::4 color 200
[*DeviceC-segment-routing-ipv6-policy-policy1] candidate-path preference 100
[*DeviceC-segment-routing-ipv6-policy-policy1-path] segment-list s1
[*DeviceC-segment-routing-ipv6-policy-policy1-path] quit
[*DeviceC-segment-routing-ipv6-policy-policy1] quit
[*DeviceC-segment-routing-ipv6] quit
[*DeviceC] commit
```

# Configure DeviceD.

```
[~DeviceD] segment-routing ipv6
[*DeviceD-segment-routing-ipv6] segment-list s1
[*DeviceD-segment-routing-ipv6-segment-list-s1] index 5 sid ipv6 2001:db8:333::333
[*DeviceD-segment-routing-ipv6-segment-list-s1] quit
[*DeviceD-segment-routing-ipv6] srv6-te-policy locator locator1
[*DeviceD-segment-routing-ipv6] srv6-te policy policy1 endpoint 2001:db8:3::3 color 200
[*DeviceD-segment-routing-ipv6-policy-policy1] candidate-path preference 100
[*DeviceD-segment-routing-ipv6-policy-policy1-path] segment-list s1
[*DeviceD-segment-routing-ipv6-policy-policy1-path] quit
[*DeviceD-segment-routing-ipv6-policy-policy1] quit
[*DeviceD-segment-routing-ipv6] quit
[*DeviceD] commit
```

After the configuration is complete, run the **display srv6-te policy** command to check SRv6 TE Policy information.

DeviceC is used as an example.

```
[~DeviceC] display srv6-te policy
PolicyName : policy1
```

|                                     |                                         |
|-------------------------------------|-----------------------------------------|
| Color : 200                         | Endpoint : 2001:db8:4::4                |
| TunnelId : 21                       | DelayTimerRemain : -                    |
| TunnelType : SRv6-TE Policy         | State Change Time : 2022-02-11 17:09:59 |
| Policy State : Up                   | Traffic Statistics : Disable            |
| Admin State : Up                    | BFD : Disable                           |
| Backup Hot-Standby : Disable        | Interface Name : -                      |
| Interface Index : -                 | Encapsulation Mode : Insert             |
| Interface State : -                 |                                         |
| Binding SID : -                     |                                         |
| Candidate-path Count : 1            |                                         |
| Candidate-path Preference : 100     |                                         |
| Path State : Active                 | Path Type : Primary                     |
| Protocol-Origin : Configuration(30) | Originator : 0, 0.0.0.0                 |
| Discriminator : 100                 | Binding SID : -                         |
| Groupid : 21                        | Policy Name : policy1                   |
| Template ID : 0                     | Path Verification : Disable             |
| DelayTimerRemain : -                | Network Slice ID : -                    |
| Segment-List Count : 1              |                                         |
| Segment-List : s1                   | XcIndex : 21                            |
| Segment-List ID : 21                | DelayTimerRemain : -                    |
| List State : Up                     | SuppressTimeRemain : -                  |
| Verification State : -              | Active PMTU : 9600                      |
| PMTU : 9600                         | BFD State : -                           |
| Weight : 1                          | BFD Delay Remain : -                    |
| Loop Detection State : Up           |                                         |
| Network Slice ID : -                |                                         |
| Binding SID : -                     |                                         |
| Reverse Binding SID : -             |                                         |
| SID : 2001:db8:444:444              |                                         |

### Step 8 Configure a tunnel policy to import VPN traffic.

```
Configure DeviceC.

[~DeviceC] route-policy RP1 permit node 1
[*DeviceC-route-policy] apply extcommunity color 0:200
[*DeviceC-route-policy] quit
[*DeviceC] bgp 100
[*DeviceC-bgp] ipv6-family vpnv6
[*DeviceC-bgp-af-vpnv6] peer 2001:db8:4::4 route-policy RP1 import
[*DeviceC-bgp-af-vpnv6] quit
[*DeviceC-bgp] quit
[*DeviceC] tunnel-policy tnl_policy
[*DeviceC-tunnel-policy-tnl_policy] tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
[*DeviceC-tunnel-policy-tnl_policy] quit
[*DeviceC] ip vpn-instance vrftest
[*DeviceC-vpn-instance-vrftest] ipv6-family
[*DeviceC-vpn-instance-vrftest-af-ipv6] tnl-policy tnl_policy
[*DeviceC-vpn-instance-vrftest-af-ipv6] commit
[~DeviceC-vpn-instance-vrftest-af-ipv6] quit
[~DeviceC-vpn-instance-vrftest] quit
```

# Configure DeviceD.

```
[~DeviceD] route-policy RP1 permit node 1
[*DeviceD-route-policy] apply extcommunity color 0:200
[*DeviceD-route-policy] quit
[*DeviceD] bgp 100
[*DeviceD-bgp] ipv6-family vpnv6
[*DeviceD-bgp-af-vpnv6] peer 2001:db8:3::3 route-policy RP1 import
[*DeviceD-bgp-af-vpnv6] quit
[*DeviceD-bgp] quit
[*DeviceD] tunnel-policy tnl_policy
[*DeviceD-tunnel-policy-tnl_policy] tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
[*DeviceD-tunnel-policy-tnl_policy] quit
[*DeviceD] ip vpn-instance vrftest
[*DeviceD-vpn-instance-vrftest] ipv6-family
[*DeviceD-vpn-instance-vrftest-af-ipv6] tnl-policy tnl_policy
```

```
[*DeviceD-vpn-instance-vrfest-af-ipv6] commit
[~DeviceD-vpn-instance-vrfest-af-ipv6] quit
[~DeviceD-vpn-instance-vrfest] quit
```

Check whether the loopback interfaces on DeviceB and DeviceE can ping each other.

DeviceB is used as an example.

```
<DeviceB>ping ipv6 -a 2001:db8:2::2 2001:db8:5::5
PING 2001:db8:5::5 : 56 data bytes, press CTRL_C to break
 Reply from 2001:db8:5::5
 bytes=56 Sequence=1 hop limit=62 time=12 ms
 Reply from 2001:db8:5::5
 bytes=56 Sequence=2 hop limit=62 time=2 ms
 Reply from 2001:db8:5::5
 bytes=56 Sequence=3 hop limit=62 time=2 ms
 Reply from 2001:db8:5::5
 bytes=56 Sequence=4 hop limit=62 time=2 ms
 Reply from 2001:db8:5::5
 bytes=56 Sequence=5 hop limit=62 time=2 ms

--- 2001:db8:5::5 ping statistics---
 5 packet(s) transmitted
 5 packet(s) received
 0.00% packet loss
 round-trip min/avg/max=2/4/12 ms
```

After the configuration is complete, run the **display ipv6 routing-table vpn-instance vrfest** command to check the routing table of the VPN instance. The command output shows that VPN routes have successfully recursed to the SRv6 TE Policy.

DeviceC is used as an example.

```
[~DeviceC] display ipv6 routing-table vpn-instance vrfest
Routing Table : vrfest
 Destinations : 5 Routes : 5

Destination : 2001:db8:2::2 PrefixLength : 128
NextHop : 2001:db8:11::1 Preference : 255
Cost : 0 Protocol : EBGP
RelayNextHop : 2001:db8:11::1 TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : RD

Destination : 2001:db8:5::5 PrefixLength : 128
NextHop : 2001:db8:4::4 Preference : 255
Cost : 0 Protocol : IBGP
RelayNextHop : :: TunnelID : 0x000000003400000015
Interface : policy1 Flags : RD

Destination : 2001:db8:11:: PrefixLength : 96
NextHop : 2001:db8:11::2 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : D

Destination : 2001:db8:11::2 PrefixLength : 128
NextHop : ::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : D

Destination : FE80:: PrefixLength : 10
NextHop : :: Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : NULL0 Flags : DB
```

```
[~DeviceC] display ipv6 routing-table vpn-instance vrfest 2001:db8:5::5 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

Routing Table : vrfest
Summary Count : 1

Destination : 2001:db8:5::5 PrefixLength : 128
NextHop : 2001:db8:4::4 Preference : 255
Neighbour : 2001:db8:4::4 ProcessID : 0
Label : 3 Protocol : IBGP
State : Active Adv Relied Cost : 0
Entry ID : 0 EntryFlags : 0x00000000
Reference Cnt: 0 Tag : 0
Priority : low Age : 224sec
IndirectID : 0x2000306 Instance :
RelayNextHop::: TunnelID : 0x000000003400000015
Interface : policy1 Flags : RD
RouteColor : 0 QoSInfo : 0x0
```

**Step 9** Establish an EBGP peer relationship between DeviceB and DeviceE.

```
Configure DeviceB.
```

```
[~DeviceB] bgp 200
[*DeviceB-bgp] peer 2001:db8:5::5 as-number 300
[*DeviceB-bgp] peer 2001:db8:5::5 ebgp-max-hop 255
[*DeviceB-bgp] peer 2001:db8:5::5 connect-interface LoopBack1
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:5::5 enable
[*DeviceB-bgp-af-ipv6] network 2001:db8:1::1 128
[*DeviceB-bgp-af-ipv6] quit
[*DeviceB-bgp] quit
[*DeviceB] commit
```

```
Configure DeviceE.
```

```
[~DeviceE] bgp 300
[*DeviceE-bgp] peer 2001:db8:2::2 as-number 200
[*DeviceE-bgp] peer 2001:db8:2::2 ebgp-max-hop 255
[*DeviceE-bgp] peer 2001:db8:2::2 connect-interface LoopBack1
[*DeviceE-bgp] ipv6-family unicast
[*DeviceE-bgp-af-ipv6] peer 2001:db8:2::2 enable
[*DeviceE-bgp-af-ipv6] quit
[*DeviceE-bgp] quit
[*DeviceE] commit
```

After the configuration is complete, run the **display bgp ipv6 peer** command on DeviceB to check whether the EBGP peer relationship between DeviceB and DeviceE has been established. If the **Established** state is displayed in the command output, the EBGP peer relationship has been established successfully.

DeviceB is used as an example.

```
[~DeviceB] display bgp ipv6 peer

BGP local router ID : 1.11.1.1
Local AS number : 200
Total number of peers : 2 Peers in established state : 2

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
2001:db8:5::5 4 300 5 6 0 00:00:24 Established 1
2001:db8:11::2 4 100 106 107 0 01:28:52 Established 1
```

**Step 10** Create a VPN instance and configure the IPv6 address family capability for it on DeviceA and DeviceE.

```
Configure DeviceA.
```

```
[~DeviceA] ip vpn-instance vrf
[*DeviceA-vpn-instance-vrf] ipv6-family
[*DeviceA-vpn-instance-vrf-af-ipv6] route-distinguisher 1:21
[*DeviceA-vpn-instance-vrf-af-ipv6] vpn-target 1:4 export-extcommunity
[*DeviceA-vpn-instance-vrf-af-ipv6] vpn-target 1:4 import-extcommunity
[*DeviceA-vpn-instance-vrf-af-ipv6] quit
[*DeviceA-vpn-instance-vrf] quit
[*DeviceA] interface loopBack 100
[*DeviceA-LoopBack100] ip binding vpn-instance vrf
[*DeviceA-LoopBack100] ipv6 enable
[*DeviceA-LoopBack100] ipv6 address 2001:db8:16::1/128
[*DeviceA-LoopBack100] quit
[*DeviceA] commit
```

# Configure DeviceE.

```
[~DeviceE] ip vpn-instance vrf
[*DeviceE-vpn-instance-vrf] ipv6-family
[*DeviceE-vpn-instance-vrf-af-ipv6] route-distinguisher 1:21
[*DeviceE-vpn-instance-vrf-af-ipv6] vpn-target 1:4 export-extcommunity
[*DeviceE-vpn-instance-vrf-af-ipv6] vpn-target 1:4 import-extcommunity
[*DeviceE-vpn-instance-vrf-af-ipv6] quit
[*DeviceE-vpn-instance-vrf] quit
[*DeviceE] interface loopBack 100
[*DeviceE-LoopBack100] ip binding vpn-instance vrf
[*DeviceE-LoopBack100] ipv6 enable
[*DeviceE-LoopBack100] ipv6 address 2001:db8:15::1/128
[*DeviceE-LoopBack100] quit
[*DeviceE] commit
```

### Step 11 Establish an EBGP peer relationship between DeviceA and DeviceE.

# Configure DeviceA.

```
[~DeviceA] bgp 200
[*DeviceA-bgp] router-id 5.55.5.5
[*DeviceA-bgp] peer 2001:db8:5::5 as-number 300
[*DeviceA-bgp] peer 2001:db8:5::5 ebgp-max-hop 255
[*DeviceA-bgp] peer 2001:db8:5::5 connect-interface LoopBack1
[*DeviceA-bgp] ipv6-family vpnv6
[*DeviceA-bgp-af-vpnv6] peer 2001:db8:5::5 enable
[*DeviceA-bgp-af-vpnv6] quit
[*DeviceA-bgp] quit
[*DeviceA] commit
```

# Configure DeviceE.

```
[~DeviceE] bgp 300
[*DeviceE-bgp] peer 2001:db8:1::1 as-number 200
[*DeviceE-bgp] peer 2001:db8:1::1 ebgp-max-hop 255
[*DeviceE-bgp] peer 2001:db8:1::1 connect-interface LoopBack1
[*DeviceE-bgp] ipv6-family vpnv6
[*DeviceE-bgp-af-vpnv6] peer 2001:db8:1::1 enable
[*DeviceE-bgp-af-vpnv6] quit
[*DeviceE-bgp] quit
[*DeviceE] commit
```

After the configuration is complete, run the **display bgp vpnv6 all peer** command on DeviceA to check whether the EBGP peer relationship between DeviceA and DeviceE has been established. If the **Established** state is displayed in the command output, the EBGP peer relationship has been established successfully.

DeviceA is used as an example.

```
[~DeviceA] display bgp vpnv6 all peer
BGP local router ID : 5.55.5.5
Local AS number : 200
Total number of peers : 1 Peers in established state : 1
```

| Peer          | V | AS  | MsgRcvd | MsgSent | OutQ | Up/Down | State       | PrefRcv |
|---------------|---|-----|---------|---------|------|---------|-------------|---------|
| 2001:db8:5::5 | 4 | 300 | 4       | 4       | 0    | 0:00:36 | Established | 1       |

**Step 12** Configure SRv6 SIDs and configure VPN routes to carry the SID attribute on DeviceA and DeviceE.

# Configure DeviceA.

```
[~DeviceA] segment-routing ipv6
[*DeviceA-segment-routing-ipv6] encapsulation source-address 2001:db8:1::1
[*DeviceA-segment-routing-ipv6] locator locator1 ipv6-prefix 2001:db8:100:: 64 static 32
[*DeviceA-segment-routing-ipv6-locator] opcode 2001:db8:100::111 end psp
[*DeviceA-segment-routing-ipv6-locator] opcode 2001:db8:100:600 end-op
[*DeviceA-segment-routing-ipv6-locator] quit
[*DeviceA-segment-routing-ipv6] quit
[*DeviceA] bgp 200
[*DeviceA-bgp] ipv6-family vpnv6
[*DeviceA-bgp-af-vpnv6] peer 2001:db8:5::5 prefix-sid
[*DeviceA-bgp-af-vpnv6] quit
[*DeviceA-bgp] ipv6-family vpn-instance vrf
[*DeviceA-bgp-6-vrf] import-route direct
[*DeviceA-bgp-6-vrf] segment-routing ipv6 traffic-engineer best-effort
[*DeviceA-bgp-6-vrf] segment-routing ipv6 locator locator1
[*DeviceA-bgp-6-vrf] quit
[*DeviceA-bgp] quit
[*DeviceA] isis 1
[*DeviceA-isis-1] segment-routing ipv6 locator locator1 auto-sid-disable
[*DeviceA-isis-1] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[~DeviceB] segment-routing ipv6
[*DeviceB-segment-routing-ipv6] encapsulation source-address 2001:db8:2::2
[*DeviceB-segment-routing-ipv6] locator locator1 ipv6-prefix 2001:db8:200:: 64 static 32
[*DeviceB-segment-routing-ipv6-locator] opcode 2001:db8:200::222 end psp
[*DeviceB-segment-routing-ipv6-locator] quit
[*DeviceB-segment-routing-ipv6] quit
[*DeviceB] isis 1
[*DeviceB-isis-1] segment-routing ipv6 locator locator1 auto-sid-disable
[*DeviceB-isis-1] quit
[*DeviceB] bgp 200
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] network 2001:db8:100:: 64
[*DeviceB-bgp-af-ipv6] network 2001:db8:200:: 64
[*DeviceB-bgp-af-ipv6] quit
[*DeviceB-bgp] quit
[*DeviceB] commit
```

# Configure DeviceE.

```
[~DeviceE] segment-routing ipv6
[*DeviceE-segment-routing-ipv6] encapsulation source-address 2001:db8:5::5
[*DeviceE-segment-routing-ipv6] locator as1 ipv6-prefix 2001:db8:555:: 64 static 32
[*DeviceE-segment-routing-ipv6-locator] opcode 2001:db8:555::555 end psp
[*DeviceE-segment-routing-ipv6-locator] opcode 2001:db8:555::500 end-op
[*DeviceE-segment-routing-ipv6-locator] quit
[*DeviceE-segment-routing-ipv6] quit
[*DeviceE] bgp 300
[*DeviceE-bgp] ipv6-family unicast
[*DeviceE-bgp-af-ipv6] network 2001:db8:555:: 64
[*DeviceE-bgp-af-ipv6] ipv6-family vpnv6
[*DeviceE-bgp-af-vpnv6] peer 2001:db8:1::1 prefix-sid
[*DeviceE-bgp-af-vpnv6] quit
[*DeviceE-bgp] ipv6-family vpn-instance vrf
[*DeviceE-bgp-6-vpna] import-route direct
[*DeviceE-bgp-6-vpna] segment-routing ipv6 traffic-engineer best-effort
[*DeviceE-bgp-6-vpna] segment-routing ipv6 locator as1
[*DeviceE-bgp-6-vpna] quit
```

```
[*DeviceE-bgp] quit
[*DeviceE] isis 2
[*DeviceE-isis-2] segment-routing ipv6 locator as1 auto-sid-disable
[*DeviceE-isis-2] quit
[*DeviceE] commit
```

Run the **display segment-routing ipv6 local-sid end forwarding** command to check information about the SRv6 local SID table.

```
[~DeviceB] display segment-routing ipv6 local-sid end forwarding

My Local-SID End Forwarding Table

SID : 2001:db8:200::222/128 FuncType : End
Flavor : PSP
LocatorName : locator1 LocatorID : 2
ProtocolType: STATIC ProcessID : --
UpdateTime : 2022-02-11 18:03:00.703

Total SID(s): 1
```

### Step 13 Configure the BGP EPE and virtual link functions on DeviceB and DeviceE.



When configuring BGP EPE, you need to enable the BGP-LS address family. Otherwise, BGP EPE cannot take effect. Route advertisement is not recommended for the virtual link between DeviceB and DeviceE.

# Configure DeviceB.

```
[~DeviceB] route-policy rp deny node 10
[*DeviceB-route-policy] quit
[*DeviceB] commit
[~DeviceB] bgp 200
[*DeviceB-bgp] peer 2001:db8:5::5 egress-engineering srv6 locator locator1
[*DeviceB-bgp] peer 2001:db8:5::5 virtual-link
[*DeviceB-bgp] peer 2001:db8:5::5 virtual-link te metric 16777215
[*DeviceB-bgp] peer 2001:db8:5::5 virtual-link te link administrative group ff
[*DeviceB-bgp] peer 2001:db8:5::5 virtual-link te srlg 4294967295
[*DeviceB-bgp] link-state-family unicast
[*DeviceB-bgp-af-ls] quit
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:5::5 route-policy rp export
[*DeviceB-bgp-af-ipv6] quit
[*DeviceB-bgp] quit
[*DeviceB] commit
```

# Configure DeviceE.

```
[~DeviceE] route-policy rp deny node 10
[*DeviceE-route-policy] quit
[*DeviceE] commit
[~DeviceE] bgp 300
[*DeviceE-bgp] peer 2001:db8:2::2 egress-engineering srv6 locator as1
[*DeviceE-bgp] peer 2001:db8:2::2 virtual-link
[*DeviceE-bgp] peer 2001:db8:2::2 virtual-link te metric 16777215
[*DeviceE-bgp] peer 2001:db8:2::2 virtual-link te link administrative group ff
[*DeviceE-bgp] peer 2001:db8:2::2 virtual-link te srlg 4294967295
[*DeviceE-bgp] link-state-family unicast
[*DeviceE-bgp-af-ls] quit
[*DeviceE-bgp] ipv6-family unicast
[*DeviceE-bgp-af-ipv6] peer 2001:db8:2::2 route-policy rp export
[*DeviceE-bgp-af-ipv6] quit
[*DeviceE-bgp] quit
[*DeviceE] commit
```

After the configuration is complete, run the **display bgp egress-engineering** command to check BGP EPE information. The command output shows SRv6 SID information.

```
[~DeviceB] display bgp egress-engineering
Peer Node : 2001:db8:5::5
Peer Adj Num : 1
Local ASN : 200
Remote ASN : 300
Local Router Id : 1.1.1.1
Remote Router Id : 4.4.4.4
Local Interface Address : 2001:db8:2::2
Remote Interface Address : 2001:db8:5::5
Usable Locator : locator1
SRv6 SID : 2001:db8:200::1:0:0
SRv6 SID (PSP) : 2001:db8:200::1:0:1
SRv6 SID (PSP,USP,USD) : 2001:db8:200::1:0:2
SRv6 SID (PSP,USP,USD,COC) : -(ONLY SUPPORT COMPRESS TYPE)
Nexthop : 2001:db8:11::2
Out Interface : GigabitEthernet1/0/0

Peer Adj : 2001:db8:11::2
Local ASN : 200
Remote ASN : 300
Local Router Id : 1.1.1.1
Remote Router Id : 4.4.4.4
Interface Identifier : 24
Local Interface Address : 2001:db8:11::1
Remote Interface Address : 2001:db8:11::2
Usable Locator : locator1
SRv6 SID : 2001:db8:200::1:0:3
SRv6 SID (PSP) : 2001:db8:200::1:0:4
SRv6 SID (PSP,USP,USD) : 2001:db8:200::1:0:5
SRv6 SID (PSP,USP,USD,COC) : -(ONLY SUPPORT COMPRESS TYPE)
Nexthop : 2001:db8:11::2
Out Interface : GigabitEthernet1/0/0
[~DeviceE] display bgp egress-engineering
Peer Node : 2001:db8:2::2
Peer Adj Num : 1
Local ASN : 300
Remote ASN : 200
Local Router Id : 4.4.4.4
Remote Router Id : 1.1.1.1
Local Interface Address : 2001:db8:5::5
Remote Interface Address : 2001:db8:2::2
Usable Locator : as1
SRv6 SID : 2001:db8:555::1:0:1
SRv6 SID (PSP) : 2001:db8:555::1:0:2
SRv6 SID (PSP,USP,USD) : 2001:db8:555::1:0:3
SRv6 SID (PSP,USP,USD,COC) : -(ONLY SUPPORT COMPRESS TYPE)
Nexthop : 2001:db8:33::1
Out Interface : GigabitEthernet2/0/0

Peer Adj : 2001:db8:33::1
Local ASN : 300
Remote ASN : 200
Local Router Id : 4.4.4.4
Remote Router Id : 1.1.1.1
Interface Identifier : 17
Local Interface Address : 2001:db8:33::2
Remote Interface Address : 2001:db8:33::1
Usable Locator : as1
SRv6 SID : 2001:db8:555::1:0:4
SRv6 SID (PSP) : 2001:db8:555::1:0:5
SRv6 SID (PSP,USP,USD) : 2001:db8:555::1:0:6
SRv6 SID (PSP,USP,USD,COC) : -(ONLY SUPPORT COMPRESS TYPE)
```

|               |   |                      |
|---------------|---|----------------------|
| Nexthop       | : | 2001:db8:33::1       |
| Out Interface | : | GigabitEthernet2/0/0 |

#### Step 14 Configure an SRv6 TE Policy.

# Configure DeviceA.

```
[~DeviceA] segment-routing ipv6
[*DeviceA-segment-routing-ipv6] segment-list list1
[*DeviceA-segment-routing-ipv6-segment-list-list1] index 5 sid ipv6 2001:db8:100::111
[*DeviceA-segment-routing-ipv6-segment-list-list1] index 10 sid ipv6 2001:db8:200::222
[*DeviceA-segment-routing-ipv6-segment-list-list1] index 15 sid ipv6 2001:db8:200::1:0:0
[*DeviceA-segment-routing-ipv6-segment-list-list1] quit
[*DeviceA-segment-routing-ipv6] srv6-te-policy locator locator1
[*DeviceA-segment-routing-ipv6] srv6-te policy policy1 endpoint 2001:db8:5::5 color 100
[*DeviceA-segment-routing-ipv6-policy-policy1] candidate-path preference 100
[*DeviceA-segment-routing-ipv6-policy-policy1-path] segment-list list1
[*DeviceA-segment-routing-ipv6-policy-policy1-path] quit
[*DeviceA-segment-routing-ipv6-policy-policy1] quit
[*DeviceA-segment-routing-ipv6] quit
[*DeviceA] commit
```

# Configure DeviceB.

```
[~DeviceB] segment-routing ipv6
[*DeviceB-segment-routing-ipv6] srv6-te-policy locator locator1
[*DeviceB-segment-routing-ipv6] quit
[*DeviceB] commit
```

# Configure DeviceE.

```
[~DeviceE] segment-routing ipv6
[*DeviceE-segment-routing-ipv6] segment-list list1
[*DeviceE-segment-routing-ipv6-segment-list-list1] index 10 sid ipv6 2001:db8:555::1:0:1
[*DeviceE-segment-routing-ipv6-segment-list-list1] index 15 sid ipv6 2001:db8:200::222
[*DeviceE-segment-routing-ipv6-segment-list-list1] index 20 sid ipv6 2001:db8:100::111
[*DeviceE-segment-routing-ipv6-segment-list-list1] quit
[*DeviceE-segment-routing-ipv6] srv6-te-policy locator as1
[*DeviceE-segment-routing-ipv6] srv6-te policy policy1 endpoint 2001:db8:1::1 color 100
[*DeviceE-segment-routing-ipv6-policy-policy1] candidate-path preference 100
[*DeviceE-segment-routing-ipv6-policy-policy1-path] segment-list list1
[*DeviceE-segment-routing-ipv6-policy-policy1-path] quit
[*DeviceE-segment-routing-ipv6-policy-policy1] quit
[*DeviceE-segment-routing-ipv6] quit
[*DeviceE] commit
```

After the configuration is complete, run the **ping srv6-te policy** command to check the connectivity of the SRv6 TE Policy.

```
<DeviceA>ping srv6-te policy policy-name policy1 end-op 2001:db8:555::500
PING srv6-te policy : 100 data bytes, press CTRL_C to break
srv6-te policy's segment list:
Preference: 100; Path Type: primary; Protocol-Origin: local; Originator: 0, 0.0.0.0; Discriminator: 100;
Segment-List ID: 129; Xcindex: 129; end-op: 2001:db8:555::500
 Reply from 2001:db8:555::500
 bytes=100 Sequence=1 time=1 ms
 Reply from 2001:db8:555::500
 bytes=100 Sequence=2 time=1 ms
 Reply from 2001:db8:555::500
 bytes=100 Sequence=3 time=1 ms
 Reply from 2001:db8:555::500
 bytes=100 Sequence=4 time=1 ms
 Reply from 2001:db8:555::500
 bytes=100 Sequence=5 time=1 ms

--- srv6-te policy ping statistics ---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max = 1/1/1 ms
```

After the configuration is complete, run the **display srv6-te policy** command to check SRv6 TE Policy information.

DeviceA is used as an example.

```
[~DeviceA] display srv6-te policy
PolicyName : policy1
Color : 100 Endpoint : 2001:db8:5::5
TunnelId : 65
TunnelType : SRv6-TE Policy
Policy State : Up DelayTimerRemain : -
Admin State : Up State Change Time : 2022-02-11 18:15:17
Backup Hot-Standby : Disable Traffic Statistics : Disable
Interface Index : - BFD : Disable
Interface State : - Interface Name : -
Binding SID : - Encapsulation Mode : Insert
Candidate-path Count : 1

Candidate-path Preference : 100
Path State : Active Path Type : Primary
Protocol-Origin : Configuration(30) Originator : 0, 0.0.0.0
Discriminator : 100 Binding SID : -
GroupId : 65 Policy Name : policy1
Template ID : 0 Path Verification : Disable
DelayTimerRemain : - Network Slice ID : -
Segment-List Count : 1
Segment-List : list1
Segment-List ID : 129 XcIndex : 129
List State : Up DelayTimerRemain : -
Verification State : - SuppressTimeRemain : -
PMTU : 9600 Active PMTU : 9600
Weight : 1 BFD State : -
Network Slice ID : - -
Binding SID : - -
Reverse Binding SID : - -
SID :
 2001:db8:100::111
 2001:db8:200::222
 2001:db8:200::1:0:0
```

### Step 15 Configure a tunnel policy to import VPN traffic.

# Configure DeviceA.

```
[~DeviceA] route-policy RP1 permit node 1
[*DeviceA-route-policy] apply extcommunity color 0:100
[*DeviceA-route-policy] quit
[*DeviceA] bgp 200
[*DeviceA-bgp] ipv6-family vpnv6
[*DeviceA-bgp-af-vpnv6] peer 2001:db8:5::5 route-policy RP1 import
[*DeviceA-bgp-af-vpnv6] quit
[*DeviceA-bgp] quit
[*DeviceA] tunnel-policy tnl_policy
[*DeviceA-tunnel-policy-tnl_policy] tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
[*DeviceA-tunnel-policy-tnl_policy] quit
[*DeviceA] ip vpn-instance vrf
[*DeviceA-vpn-instance-vrf] ipv6-family
[*DeviceA-vpn-instance-vrf-af-ipv6] tnl-policy tnl_policy
[*DeviceA-vpn-instance-vrf-af-ipv6] commit
[~DeviceA-vpn-instance-vrf-af-ipv6] quit
[~DeviceA-vpn-instance-vrf] quit
```

# Configure DeviceE.

```
[~DeviceE] route-policy RP1 permit node 1
[*DeviceE-route-policy] apply extcommunity color 0:100
[*DeviceE-route-policy] quit
[*DeviceE] bgp 300
[*DeviceE-bgp] ipv6-family vpnv6
```

```
[*DeviceE-bgp-af-vpnv6] peer 2001:db8:1::1 route-policy RP1 import
[*DeviceE-bgp-af-vpnv6] quit
[*DeviceE-bgp] quit
[*DeviceE] tunnel-policy tnl_policy
[*DeviceE-tunnel-policy-tnl_policy] tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
[*DeviceE-tunnel-policy-tnl_policy] quit
[*DeviceE] ip vpn-instance vrf
[*DeviceE-vpn-instance-vrf] ipv6-family
[*DeviceE-vpn-instance-vrf-af-ipv6] tnl-policy tnl_policy
[*DeviceE-vpn-instance-vrf-af-ipv6] commit
[~DeviceE-vpn-instance-vrf-af-ipv6] quit
[~DeviceE-vpn-instance-vrf] quit
```

After the configuration is complete, run the **display ipv6 routing-table vpn-instance vrf** command to check the routing table of the VPN instance. The command output shows that VPN routes have successfully recursed to the SRv6 TE Policy.

DeviceA is used as an example.

```
[~DeviceA] display ipv6 routing-table vpn-instance vrf
Routing Table : vrf
Destinations : 3 Routes : 3

Destination : 2001:db8:15::1 PrefixLength : 128
NextHop : 2001:db8:5::5 Preference : 255
Cost : 0 Protocol : EBGP
RelayNextHop : :: TunnelID : 0x000000003400000041
Interface : policy1 Flags : RD

Destination : 2001:db8:16::1 PrefixLength : 128
NextHop : 2001:db8:11::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : LoopBack100 Flags : D

Destination : FE80:: PrefixLength : 10
NextHop : :: Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : NULL0 Flags : DB
[~DeviceA] display ipv6 routing-table vpn-instance vrf 2001:db8:15::1 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route

```

```
Routing Table : vrf
Summary Count : 1

Destination : 2001:db8:15::1 PrefixLength : 128
NextHop : 2001:db8:5::5 Preference : 255
Neighbour : 2001:db8:5::5 ProcessID : 0
Label : 3 Protocol : EBGP
State : Active Adv Relied Cost : 0
Entry ID : 0 EntryFlags : 0x00000000
Reference Cnt: 0 Tag : 0
Priority : low Age : 117sec
IndirectID : 0x100049D Instance :
RelayNextHop : :: TunnelID : 0x000000003400000041
Interface : policy1 Flags : RD
RouteColor : 0 QoSInfo : 0x0
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA
#
```

```
ip vpn-instance vrf
 ipv6-family
 route-distinguisher 1:21
 tnl-policy tnl_policy
 apply-label per-instance
 vpn-target 1:4 export-extcommunity
 vpn-target 1:4 import-extcommunity
#
segment-routing ipv6
 encapsulation source-address 2001:db8:1::1
 locator locator1 ipv6-prefix 2001:db8:100:: 64 static 32
 opcode 2001:db8:100::111 end psp
 opcode 2001:db8:100:600 end-op
 srv6-te-policy locator locator1
 segment-list list1
 index 5 sid ipv6 2001:db8:100::111
 index 10 sid ipv6 2001:db8:200::222
 index 15 sid ipv6 2001:db8:200::1:0:0
 srv6-te policy policy1 endpoint 2001:db8:5::5 color 100
 candidate-path preference 100
 segment-list list1
#
isis 1
 is-level level-1
 cost-style wide
 network-entity 10.0000.0000.0001.00
#
 ipv6 enable topology ipv6
 segment-routing ipv6 locator locator1 auto-sid-disable
#
#
 interface GigabitEthernet1/0/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:44::1/96
 isis ipv6 enable 1
 undo dcn
#
 interface LoopBack 1
 ipv6 enable
 ipv6 address 2001:db8:1::1/128
 isis ipv6 enable 1
#
 interface LoopBack100
 ip binding vpn-instance vrf
 ipv6 enable
 ipv6 address 2001:db8:16::1/128
#
 bgp 200
 router-id 5.55.5.5
 private-4-byte-as enable
 peer 2001:db8:5::5 as-number 300
 peer 2001:db8:5::5 ebgp-max-hop 255
 peer 2001:db8:5::5 connect-interface LoopBack1
#
 ipv4-family unicast
 undo synchronization
#
 ipv6-family vpnv6
 policy vpn-target
 peer 2001:db8:5::5 enable
 peer 2001:db8:5::5 route-policy RP1 import
 peer 2001:db8:5::5 prefix-sid
#
 ipv6-family vpn-instance vrf
 import-route direct
 segment-routing ipv6 locator locator1
 segment-routing ipv6 traffic-engineer best-effort
#
```

```
route-policy RP1 permit node 1
 apply extcommunity color 0:100
#
tunnel-policy tnl_policy
 tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
segment-routing ipv6
 encapsulation source-address 2001:db8:2::2
 locator locator1 ipv6-prefix 2001:db8:200:: 64 static 32
 opcode 2001:db8:200::222 end psp
 srv6-te-policy locator locator1
#
isis 1
 is-level level-1
 cost-style wide
 network-entity 10.0000.0000.0002.00
#
 ipv6 enable topology ipv6
 segment-routing ipv6 locator locator1 auto-sid-disable
 ipv6 import-route bgp level-1
#
#
interface GigabitEthernet1/0/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:11::1/96
#
interface GigabitEthernet2/0/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:44::2/96
 isis ipv6 enable 1
#
interface LoopBack1
 ipv6 enable
 ipv6 address 2001:db8:2::2/128
#
bgp 200
 router-id 1.1.1.1
 private-4-byte-as enable
 peer 2001:db8:5::5 as-number 300
 peer 2001:db8:5::5 ebgp-max-hop 255
 peer 2001:db8:5::5 connect-interface LoopBack1
 peer 2001:db8:5::5 egress-engineering srv6 locator locator1
 peer 2001:db8:5::5 virtual-link
 peer 2001:db8:5::5 virtual-link te metric 16777215
 peer 2001:db8:5::5 virtual-link te link administrative group ff
 peer 2001:db8:5::5 virtual-link te srlg 4294967295
 peer 2001:db8:11::2 as-number 100
#
 ipv4-family unicast
 undo synchronization
#
 link-state-family unicast
#
 ipv6-family unicast
 undo synchronization
 network 2001:db8:1::1 128
 network 2001:db8:2::2 128
 network 2001:db8:100:: 64
 network 2001:db8:200:: 64
 peer 2001:db8:5::5 enable
 peer 2001:db8:5::5 route-policy rp export
 peer 2001:db8:11::2 enable
```

```

route-policy rp deny node 10

return
```

- DeviceC configuration file

```

sysname DeviceC

ip vpn-instance vrftest
ipv6-family
 route-distinguisher 1:1
 tnl-policy tnl_policy
 apply-label per-instance
 vpn-target 100:1 export-extcommunity
 vpn-target 100:1 import-extcommunity

segment-routing ipv6
encapsulation source-address 2001:db8:3::3
locator locator1 ipv6-prefix 2001:db8:333:: 64 static 32
 opcode 2001:db8:333::333 end psp
 srv6-te-policy locator locator1
segment-list s1
 index 5 sid ipv6 2001:db8:444::444
 srv6-te policy policy1 endpoint 2001:db8:4::4 color 200
 candidate-path preference 100
 segment-list s1

isis 100
is-level level-2
cost-style wide
network-entity 20.0000.0000.0001.00

ipv6 enable topology ipv6
segment-routing ipv6 locator locator1 auto-sid-disable

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:22::1/96
isis ipv6 enable 100

interface GigabitEthernet2/0/0
undo shutdown
ip binding vpn-instance vrftest
ipv6 enable
ipv6 address 2001:db8:11::2/96

interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:3::3/128
isis ipv6 enable 100

bgp 100
router-id 2.22.2.2
private-4-byte-as enable
peer 2001:db8:4::4 as-number 100
peer 2001:db8:4::4 connect-interface LoopBack1

ipv4-family unicast
undo synchronization

ipv6-family unicast
undo synchronization

ipv6-family vpng6
policy vpn-target
peer 2001:db8:4::4 enable
peer 2001:db8:4::4 route-policy RP1 import
```

```
peer 2001:db8:4::4 prefix-sid
#
ipv6-family vpn-instance vrftest
segment-routing ipv6 locator locator1
segment-routing ipv6 traffic-engineer best-effort
peer 2001:db8:11::1 as-number 200
#
route-policy RP1 permit node 1
apply extcommunity color 0:200
#
tunnel-policy tnl_policy
tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
ip vpn-instance vrftest
ipv6-family
route-distinguisher 1:1
tnl-policy tnl_policy
apply-label per-instance
vpn-target 100:1 export-extcommunity
vpn-target 100:1 import-extcommunity
#
segment-routing ipv6
encapsulation source-address 2001:db8:4::4
locator locator1 ipv6-prefix 2001:db8:444:: 64 static 32
opcode 2001:db8:444::444 end psp
srv6-te-policy locator locator1
segment-list s1
index 5 sid ipv6 2001:db8:333::333
srv6-te policy policy1 endpoint 2001:db8:3::3 color 200
candidate-path preference 100
segment-list s1
#
isis 100
is-level level-2
cost-style wide
network-entity 20.0000.0000.0002.00
#
ipv6 enable topology ipv6
segment-routing ipv6 locator locator1 auto-sid-disable
#
#
interface GigabitEthernet1/0/0
undo shutdown
ip binding vpn-instance vrftest
ipv6 enable
ipv6 address 2001:db8:33::1/96
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:22::2/96
isis ipv6 enable 100
#
interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:4::4/128
isis ipv6 enable 100
#
bgp 100
router-id 3.33.3.3
private-4-byte-as enable
peer 2001:db8:3::3 as-number 100
peer 2001:db8:3::3 connect-interface LoopBack1
#
```

```
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
#
ipv6-family vpng6
policy vpn-target
peer 2001:db8:3::3 enable
peer 2001:db8:3::3 route-policy RP1 import
peer 2001:db8:3::3 prefix-sid
#
ipv6-family vpn-instance vrftest
segment-routing ipv6 locator locator1
segment-routing ipv6 traffic-engineer best-effort
peer 2001:db8:33::2 as-number 300
#
route-policy RP1 permit node 1
apply extcommunity color 0:200
#
tunnel-policy tnl_policy
tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
ip vpn-instance vrf
ipv6-family
route-distinguisher 1:21
tnl-policy tnl_policy
apply-label per-instance
vpn-target 1:4 export-extcommunity
vpn-target 1:4 import-extcommunity
#
segment-routing ipv6
encapsulation source-address 2001:db8:5::5
locator as1 ipv6-prefix 2001:db8:555:: 64 static 32
opcode 2001:db8:555::555 end psp
opcode 2001:db8:555::500 end-op
srv6-te-policy locator as1
segment-list list1
index 10 sid ipv6 2001:db8:555::1:0:1
index 15 sid ipv6 2001:db8:200::222
index 20 sid ipv6 2001:db8:100::111
srv6-te policy policy1 endpoint 2001:db8:1::1 color 100
candidate-path preference 100
segment-list list1
#
isis 2
#
segment-routing ipv6 locator as1 auto-sid-disable
#
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:33::2/96
#
interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:5::5/128
#
interface LoopBack100
ip binding vpn-instance vrf
ipv6 enable
ipv6 address 2001:db8:15::1/128
#
```

```
bgp 300
router-id 4.44.4.4
private-4-byte-as enable
peer 2001:db8:1::1 as-number 200
peer 2001:db8:1::1 ebgp-max-hop 255
peer 2001:db8:1::1 connect-interface LoopBack1
peer 2001:db8:2::2 as-number 200
peer 2001:db8:2::2 ebgp-max-hop 255
peer 2001:db8:2::2 connect-interface LoopBack1
peer 2001:db8:2::2 egress-engineering srv6 locator as1
peer 2001:db8:2::2 virtual-link
peer 2001:db8:2::2 virtual-link te metric 16777215
peer 2001:db8:2::2 virtual-link te link administrative group ff
peer 2001:db8:2::2 virtual-link te srlg 4294967295
peer 2001:db8:33::1 as-number 100
#
ipv4-family unicast
undo synchronization
#
link-state-family unicast
#
ipv6-family unicast
undo synchronization
network 2001:db8:5::5 128
network 2001:db8:555:: 64
peer 2001:db8:2::2 enable
peer 2001:db8:2::2 route-policy rp export
peer 2001:db8:33::1 enable
#
ipv6-family vpng6
policy vpn-target
peer 2001:db8:1::1 enable
peer 2001:db8:1::1 route-policy RP1 import
peer 2001:db8:1::1 prefix-sid
#
ipv6-family vpn-instance vrf
import-route direct
segment-routing ipv6 locator as1
segment-routing ipv6 traffic-engineer best-effort
#
route-policy RP1 permit node 1
apply extcommunity color 0:100
#
route-policy rp deny node 10
#
tunnel-policy tnl_policy
tunnel select-seq ipv6 srv6-te-policy load-balance-number 1
#
return
```

## Example for Configuring BGP Routing Loop Detection

This section describes how to configure BGP routing loop detection to detect potential routing loops on a network.

## Networking Requirements

On the network shown in [Figure 1-448](#), DeviceA, DeviceB, and DeviceC belong to AS 100. An IBGP peer relationship is established between DeviceA and the RR, between the RR and DeviceB, and between the RR and DeviceC. OSPF runs on DeviceB and DeviceC. DeviceB is configured to import BGP routes to OSPF, and DeviceC is configured to import OSPF routes to BGP. An export policy is configured on DeviceA to add AS numbers to the AS\_Path attribute for the routes to be advertised to the RR. After receiving a BGP route from DeviceA, the RR advertises this route to DeviceB. DeviceB then imports the BGP route to convert it to an OSPF

route and advertises the OSPF route to DeviceC. DeviceC then imports the OSPF route to convert it to a BGP route and advertises the BGP route to the RR. When comparing the route advertised by DeviceA and the route advertised by DeviceC, the RR prefers the one advertised by DeviceC because the AS\_Path of the route advertised by DeviceA is longer than that of the route advertised by DeviceC. As a result, a stable routing loop occurs.

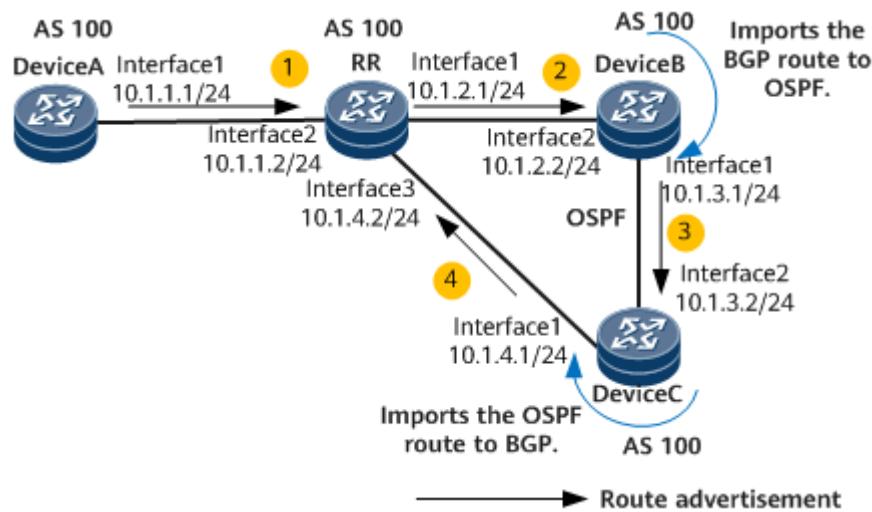
To address this problem, enable BGP routing loop detection on DeviceC. After BGP routing loop detection is enabled, DeviceC adds Loop-detection attribute 1 to the BGP route imported from OSPF and advertises the BGP route to the RR. After receiving this BGP route, the RR advertises it (carrying Loop-detection attribute 1) to DeviceB. As OSPF routing loop detection is enabled by default, when the BGP route is imported to become an OSPF route on DeviceB, the OSPF route inherits the routing loop attribute of the BGP route and has an OSPF routing loop attribute added as well before the OSPF route is advertised to DeviceC. Upon receipt of the OSPF route, DeviceC imports it to convert it to a BGP route. Because BGP routing loop detection is enabled, the BGP route inherits the routing loop attributes of the OSPF route. Upon receipt of the route, DeviceC finds that the received route carries its own routing loop attribute and therefore determines that a routing loop has occurred. In this case, DeviceC generates an alarm, and reduces the local preference and increases the MED value of the route before advertising the route to the RR. After receiving the route, the RR compares this route with the route advertised by DeviceA. Because the route advertised by DeviceC has a lower local preference and a larger MED value, the RR preferentially selects the route advertised by DeviceA. The routing loop is then resolved.

When the OSPF route is transmitted to DeviceC again, DeviceC imports it to convert it to a BGP route, and the route carries only the OSPF routing loop attribute added by DeviceB. However, DeviceC still considers the route as a looped route because the route has a routing loop record. In this case, the RR does not preferentially select the route after receiving it from DeviceC. Then routes converge normally.

 **NOTE**

Interface1, interface2, and interface3 in this example represent GE1/0/1, GE1/0/2, and GE1/0/3, respectively.

**Figure 1-448** Configuring BGP routing loop detection



## Precautions

To improve security, you are advised to deploy BGP security measures. For details, see "Configuring BGP Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IP addresses for interfaces.
2. Configure an IGP area and IBGP peers.
3. Configure route import on DeviceB and DeviceC.
4. Configure an export policy on DeviceA to add AS numbers to the AS\_Path attribute for the routes to be advertised to the RR. This configuration helps simulate a routing loop scenario.
5. Configure BGP routing loop detection on DeviceC.

## Data Preparation

To complete the configuration, you need the following data:

- IP addresses of devices A through C
- AS number of devices A through C

## Procedure

### Step 1 Configure IP addresses for interfaces.

DeviceA is used as an example.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 1/0/1
[~DeviceA-GigabitEthernet1/0/1] ip address 10.1.1.1 24
[*DeviceA-GigabitEthernet1/0/1] quit
[*DeviceA] commit
```

For other devices, see the configuration files.

### Step 2 Configure OSPF on DeviceB and DeviceC to implement interworking in the IGP area.

# Configure DeviceB.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 10.1.3.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
[~DeviceB-ospf-1] quit
```

# Configure DeviceC.

```
[~DeviceC] ospf 1
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 10.1.3.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
[~DeviceC-ospf-1] quit
```

**Step 3** Establish an IBGP peer relationship between DeviceA and the RR, between the RR and DeviceB, and between the RR and DeviceC.

# Configure DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 10.1.1.2 as-number 100
[*DeviceA-bgp] quit
[*DeviceA] commit
```

# Configure the RR.

```
[~RR] bgp 100
[*RR-bgp] router-id 2.2.2.2
[*RR-bgp] peer 10.1.1.1 as-number 100
[*RR-bgp] peer 10.1.2.2 as-number 100
[*RR-bgp] peer 10.1.4.1 as-number 100
[*RR-bgp] ipv4-family unicast
[*RR-bgp-af-ipv4] peer 10.1.2.2 reflect-client
[*RR-bgp-af-ipv4] quit
[*RR-bgp] quit
[*RR] commit
```

# Configure DeviceB.

```
[~DeviceB] bgp 100
[*DeviceB-bgp] router-id 3.3.3.3
[*DeviceB-bgp] peer 10.1.2.1 as-number 100
[*DeviceB-bgp] quit
[*DeviceB] commit
```

# Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] router-id 4.4.4.4
[*DeviceC-bgp] peer 10.1.4.2 as-number 100
[*DeviceC-bgp] quit
[*DeviceC] commit
```

**Step 4** Configure BGP to import static routes and direct routes.

# Configure DeviceA to import static routes.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] import-route static
[*DeviceA-bgp-af-ipv4] quit
[*DeviceA-bgp] quit
[*DeviceA] commit
```

# Configure the RR to import direct routes.

```
[~RR] bgp 100
[*RR-bgp] ipv4-family unicast
[*RR-bgp-af-ipv4] import-route direct
[*RR-bgp-af-ipv4] quit
[*RR-bgp] quit
[*RR] commit
```

**Step 5** Configure DeviceB to import BGP routes into OSPF.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] import-route bgp permit-ibgp
[*DeviceB-ospf-1] commit
[~DeviceB-ospf-1] quit
```

**Step 6** Configure DeviceC to import OSPF routes into BGP.

# Configure DeviceC.

```
[~DeviceC] bgp 100
[*DeviceC-bgp] import-route ospf 1
[*DeviceC-bgp] quit
[*DeviceC] commit
```

**Step 7** Configure a route-policy.

```
Configure DeviceA to add AS numbers to the AS_Path attribute for the routes to
be advertised.
```

```
[~DeviceA] route-policy ex1 permit node 10
[*DeviceA-route-policy] apply as-path 700 8000 additive
[*DeviceA-route-policy] quit
[*DeviceA] commit
[~DeviceA] bgp 100
[*DeviceA-bgp] ipv4-family unicast
[*DeviceA-bgp-af-ipv4] peer 10.1.1.2 route-policy ex1 export
[*DeviceA-bgp-af-ipv4] quit
[*DeviceA-bgp] quit
[*DeviceA] commit
```

**Step 8** Configure a static route.

```
Configure DeviceA to simulate a looped route.
```

```
[~DeviceA] ip route-static 10.7.7.7 255.255.255.255 NULL0
[*DeviceA] commit
```

**Step 9** Verify the configuration.

After the configuration is complete, check the BGP routing table on the RR. The command output shows that the RR has preferentially selected the route received from DeviceC and that a stable routing loop is formed. The route learned from DeviceA is not selected as the optimal route because the AS\_Path of the route learned from DeviceA is longer than that of the route learned from DeviceC.

```
<RR> display bgp routing-table 10.7.7.7

BGP local router ID : 2.2.2.2
Local AS number : 100
Paths: 2 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 10.7.7.7/32:
From: 10.1.4.1 (4.4.4.4)
Route Duration: 0d00h00m10s
Relay IP Nexthop: 10.1.4.1
Relay IP Out-Interface: GigabitEthernet1/0/3
Original nexthop: 10.1.4.1
Qos information : 0x0
AS-path Nil, origin incomplete, MED 1, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Advertised to such 1 peers:
 10.1.2.2

BGP routing table entry information of 10.7.7.7/32:
From: 10.1.1.1 (1.1.1.1)
Route Duration: 0d01h19m49s
Relay IP Nexthop: 10.1.1.1
Relay IP Out-Interface: GigabitEthernet1/0/2
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 700 8000, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, pre 255, not
preferred for AS-Path
Not advertised to any peer yet
```

**Step 10** Configure BGP routing loop detection.

```
Configure DeviceC.
```

```
[~DeviceC] route loop-detect bgp enable
[*DeviceC] commit
```

**Step 11** Verify the configuration.

After BGP routing loop detection is configured, DeviceC determines that the received route 10.7.7.7 is a looped route. Therefore, DeviceC reduces the local preference and increases the MED value of the route before advertising the route to the RR. After receiving the route, the RR compares this route with the route advertised by DeviceA. Because the route advertised by DeviceC has a lower local preference and a larger MED value, the RR preferentially selects the route advertised by DeviceA. This resolves the routing loop.

```
<RR> display bgp routing-table 10.7.7.7

BGP local router ID : 2.2.2.2
Local AS number : 100
Paths: 2 available, 1 best, 1 select, 0 best-external, 0 add-path
BGP routing table entry information of 10.7.7.7/32:
From: 10.1.1.1 (1.1.1.1)
Route Duration: 0d01h21m03s
Relay IP Nexthop: 10.1.1.1
Relay IP Out-Interface: GigabitEthernet1/0/2
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 700 8000, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Advertised to such 1 peers:
 10.1.2.2

BGP routing table entry information of 10.7.7.7/32:
From: 10.1.4.1 (4.4.4.4)
Route Duration: 0d00h00m09s
Relay IP Nexthop: 10.1.4.1
Relay IP Out-Interface: GigabitEthernet1/0/3
Original nexthop: 10.1.4.1
Qos information : 0x0
AS-path Nil, origin incomplete, MED 4294967295, localpref 0, pref-val 0, valid, internal, pre 255, not
preferred for Local_Pref
Not advertised to any peer yet
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.1.1 255.255.255.0

bgp 100
router-id 1.1.1.1
private-4-byte-as enable
peer 10.1.1.2 as-number 100

ipv4-family unicast
undo synchronization
import-route static
peer 10.1.1.2 enable
peer 10.1.1.2 route-policy ex1 export

route-policy ex1 permit node 10
apply as-path 700 8000 additive
#
```

```
ip route-static 10.7.7.7 255.255.255.255 NULL0
#
return
```

- RR configuration file

```
#
sysname RR
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet1/0/3
undo shutdown
ip address 10.1.4.2 255.255.255.0
#
bgp 100
router-id 2.2.2.2
private-4-byte-as enable
peer 10.1.1.1 as-number 100
peer 10.1.2.2 as-number 100
peer 10.1.4.1 as-number 100
#
ipv4-family unicast
undo synchronization
import-route direct
peer 10.1.1.1 enable
peer 10.1.2.2 enable
peer 10.1.2.2 reflect-client
peer 10.1.4.1 enable
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.3.1 255.255.255.0
ospf enable 1 area 0.0.0.0
#
interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.2.2 255.255.255.0
#
bgp 100
router-id 3.3.3.3
private-4-byte-as enable
peer 10.1.2.1 as-number 100
#
ipv4-family unicast
undo synchronization
peer 10.1.2.1 enable
#
ospf 1
import-route bgp permit-ibgp
opaque-capability enable
area 0.0.0.0
network 10.1.3.0 0.0.0.255
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
```

```

interface GigabitEthernet1/0/1
undo shutdown
ip address 10.1.4.1 255.255.255.0

interface GigabitEthernet1/0/2
undo shutdown
ip address 10.1.3.2 255.255.255.0
ospf enable 1 area 0.0.0.0

bgp 100
router-id 4.4.4.4
private-4-byte-as enable
peer 10.1.4.2 as-number 100

ipv4-family unicast
undo synchronization
import-route ospf 1
peer 10.1.4.2 enable

route loop-detect bgp enable

ospf 1
opaque-capability enable
area 0.0.0.0
network 10.1.3.0 0.0.0.255

return
```

## 1.1.10 BGP4+ Configuration

### 1.1.10.1 BGP4+ Configuration

BGP4+ transmits routing information between Autonomous Systems (ASs) and is applicable to large-scale and complex IPv6 networks.

#### 1.1.10.1.1 Overview of BGP4+

BGP4+ controls route advertisement and selects optimal routes.

#### BGP4+ Definition

As a dynamic routing protocol used between ASs, BGP4+ is an extension of BGP.

Traditional BGP4 manages IPv4 routing information but does not support the inter-AS transmission of packets encapsulated by other network layer protocols (such as IPv6).

To support IPv6, BGP4 must have the additional ability to associate an IPv6 protocol with the next hop information and network layer reachable information (NLRI).

Two NLRI attributes that were introduced to BGP4+ are as follows:

- Multiprotocol Reachable NLRI (MP\_REACH\_NLRI): carries the set of reachable destinations and the next hop information.

**Figure 1-449 MP\_REACH\_NLRI structure**

|                                                   |
|---------------------------------------------------|
| AFI (2 octets)                                    |
| SAFI (1 octet)                                    |
| Length of Next Hop Network Address (1 octet)      |
| Network Address of Next Hop (variable)            |
| Number of SNPAs (1 octet)                         |
| Network Layer Reachability Information (variable) |

**Table 1-156 Fields in MP\_REACH\_NLRI**

| Field                                       | Length   | Description                                                                                                                                                                                     |
|---------------------------------------------|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| AFI (address family identifier)             | 2 bytes  | Indicates the address class to which the network layer protocol belongs. It is used to specify the carried IPv6 reachable route information.                                                    |
| SAFI (subsequent address family identifier) | 1 byte   | Indicates that the attribute carries information about reachable IPv6 unicast routes.                                                                                                           |
| Length of Next Hop Network Address          | 1 byte   | Indicates the length of bytes occupied by the next hop. The value 16 indicates that the link-local address is not included, and the value 32 indicates that the link-local address is included. |
| Network Address of Next Hop                 | Variable | Indicates information about the next-hop address to the destination network, which may contain a link-local address.                                                                            |
| Number of SNPAs                             | 1 byte   | Indicates a reserved bit. The value is 0.                                                                                                                                                       |
| Network Layer Reachability Information      | Variable | Indicates carried IPv6 reachable route information, including the IPv6 prefix.                                                                                                                  |

- Multiprotocol Unreachable NLRI (MP\_UNREACH\_NLRI): carries the set of unreachable destinations.

**Figure 1-450 MP\_UNREACH\_NLRI structure**

|                                                    |
|----------------------------------------------------|
| AFI (2 octets)                                     |
| SAFI (1 octet)                                     |
| Network Layer UnReachability Information (1 octet) |

**Table 1-157 Fields in MP\_UNREACH\_NLRI**

| Field                                       | Length  | Description                                                                             |
|---------------------------------------------|---------|-----------------------------------------------------------------------------------------|
| AFI (Address Family Identifier)             | 2 bytes | Indicates that the attribute carries information about unreachable IPv6 unicast routes. |
| SAFI (subsequent address family identifier) | 1 byte  | Indicates that the attribute carries information about unreachable IPv6 unicast routes. |
| Network Layer UnReachability Information    | 1 byte  | Indicates the carried IPv6 unreachable route information.                               |

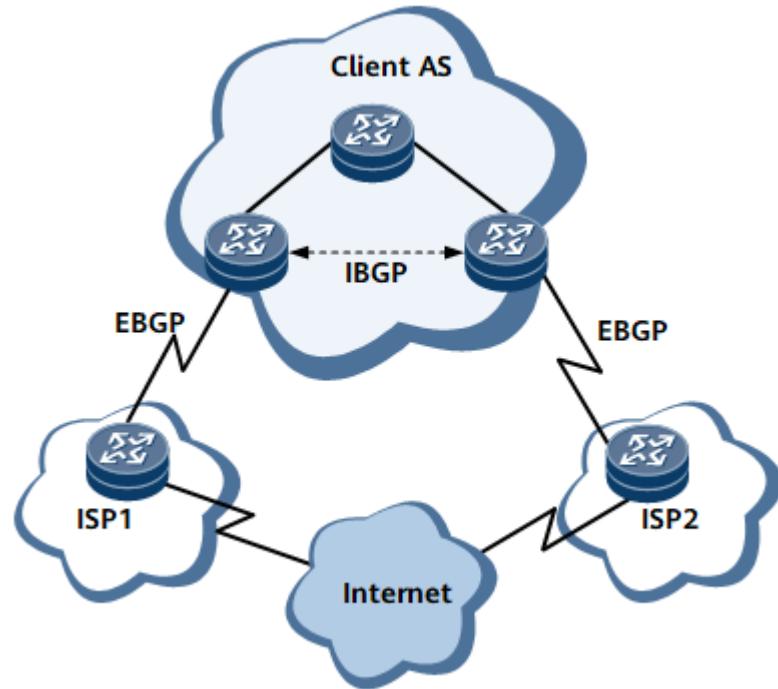
The Next\_Hop attribute in BGP4+ is in the format of an IPv6 address, which can be either a globally unique IPv6 address or a next hop link-local address.

Using multiple protocol extensions of BGP, BGP4+ is applicable to IPv6 networks without changing the messaging and routing mechanisms of BGP.

## Purpose

BGP transmits route information between ASs. It, however, is not required in all scenarios.

**Figure 1-451 BGP networking**



BGP is required in the following scenarios:

- On the network shown in [Figure 1-451](#), users need to be connected to two or more ISPs. The ISPs need to provide all or part of the Internet routes for the users. Devices, therefore, need to select the optimal route through the AS of an ISP to the destination based on the attributes carried in BGP routes.
- The AS\_Path attribute needs to be transmitted between users in different organizations.
- Users need to transmit VPN routes through a Layer 3 VPN. For details, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - VPN*.
- Users need to transmit multicast routes to construct a multicast topology. For details, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Feature Description - IP Multicast*.

BGP is not required in the following scenarios:

- Users are connected to only one ISP.
- The ISP does not need to provide Internet routes for users.
- ASs are connected through default routes.

### 1.1.10.1.2 Configuration Precautions for BGP4+

#### Feature Requirements

None

### 1.1.10.1.3 Configuring Basic BGP4+ Functions

Basic BGP4+ functions must be configured before you configure subsequent BGP4+ functions on a BGP4+ network.

#### Usage Scenario

Basic BGP4+ functions must be configured first when you configure BGP4+ for inter-AS communication.

#### Pre-configuration Tasks

Before configuring basic BGP4+ functions, configure link layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol on the interfaces is Up.

#### Starting a BGP Process

A BGP process must be started before configuring BGP functions. When starting a BGP process, specify the number of the AS to which the device belongs.

#### Context



Changing BGP router IDs will cause the reestablishment of the BGP connection between routers.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 (Optional) Run **route loop-detect bgp enable**

BGP routing loop detection is enabled.

After this function is enabled, the device reports an alarm when detecting a BGP4+ routing loop. Because the device cannot determine whether the routing loop is removed, the alarm will not be cleared automatically even if the routing loop is removed. To manually clear the BGP routing loop alarm, run the **clear route loop-detect bgp alarm** command.

### Step 3 Run **bgp as-number**

BGP is enabled (the local AS number is specified), and the BGP view is displayed.

### Step 4 (Optional) Run **router-id ipv4-address**

A BGP router ID is configured.

If two devices have different router IDs, an IBGP or EBGP connection can be established between them. If the router IDs are the same and the **router-id allow-same enable** command is run, an EBGP connection can be established.

#### NOTE

If the router ID of the router is the IP address of a physical interface, a change in the IP address will cause route flapping on the network. To enhance network stability, configuring the IP address of a loopback interface as the router ID is recommended.

If no interface on a device is configured with an IPv4 address, you need to configure a router ID for the device.

### Step 5 (Optional) Run **shutdown**

All sessions between the device and its BGP4+ peers are terminated.

During the system upgrade or maintenance, you can run this command to terminate all sessions between a device and its BGP4+ peers to prevent possible BGP4+ route flapping from affecting the network.

#### NOTICE

After the upgrade or maintenance, run the **undo shutdown** command to restore the BGP4+ peer sessions; otherwise, BGP4+ functions will be affected.

### Step 6 Run **commit**

The configuration is committed.

----End

## Configuring IPv6 Peers

Devices can exchange BGP4+ routing information only after the IPv6 peer relationship is established among them.

### Context

Because BGP4+ uses TCP connections, the IPv6 addresses for peers must be specified when you configure BGP4+. A BGP4+ peer may not be a neighboring router, and a BGP4+ peer relationship can be created by using a logical link. Using the addresses of loopback interfaces to set up BGP4+ peer relationships can improve the stability of BGP4+ connections and is recommended.

The devices in the same AS establish IBGP peer relationships, and the devices of different ASs establish EBGP peer relationships.

### Procedure

- Configure IBGP peers.

Perform the following steps on the routers between which an IBGP peer relationship needs to be set up:

- Run **system-view**

The system view is displayed.

- Run **bgp as-number**

The BGP view is displayed.

- Run **peer { ipv6-address | peerGroupName } as-number as-number**

The address of the remote peer and the AS to which the remote peer belongs are configured.

The number of the AS where the specified peer resides must be the same as that of the local AS.

The IP address of the specified peer can be one of the following types:

- IPv6 address of an interface on a directly connected peer
- IP address of a routable loopback interface on the peer
- IPv6 address of a sub-interface on a directly connected peer
- Link-local address of an interface on a directly connected peer

If the specified peer IPv6 address is the address of a loopback interface, you need to configure the local interface for the BGP4+ connection to ensure that the peer relationship is correctly established.

- (Optional) Run **peer { ipv6-address | group-name } connect-interface interface-type interface-number [ ipv6-source-address ]**

The source interface and source address for establishing a TCP connection are specified. To improve the reliability and stability of a BGP4+ connection to be established through loopback interfaces, you need to specify the local interface to be used to establish this BGP4+ connection.

 NOTE

When establishing multiple peer relationships between two routers through multiple links, you are advised to run the **peer connect-interface** command to specify the interface for establishing each BGP4+ connection.

If the source interface used by the local end to establish a BGP4+ session is configured with multiple IP addresses, the **peer connect-interface** command must be run to specify the source address. This ensures the correct connection between the two ends.

e. (Optional) Run **peer { ipv6-address | group-name } listen-only**

The local peer (group) is configured to accept connection requests, but not to send connection requests.

After this command is run, the existing peer relationship is interrupted. The local peer will wait for a connection request from the remote peer to reestablish a peer relationship. This command enables only one peer to send connection requests, preventing a connection request conflict.

 NOTE

This command can only be run on one of two peers. If this command is run on both peers, the connection between them cannot be reestablished.

f. (Optional) Run **peer { ipv6-address | group-name } description description-text**

The description of the peer is configured.

You can simplify network management by configuring the descriptions of peers.

g. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

h. Run **peer { ipv6-address | group-name } enable**

The IPv6 peer is enabled.

After configuring a BGP4+ peer in the BGP view, you also need to enable the peer in the IPv6 unicast address family view.

i. Run **commit**

The configuration is committed.

• Configure EBGP peers.

Perform the following steps on the routers between which an EBGP peer relationship needs to be set up:

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. (Optional) Run **router-id allow-same enable**

The peers with the same router ID are allowed to establish EBGP connections.

d. Run **peer { ipv6-address | group-name } as-number as-number**

The IPv6 address of the remote peer and the AS to which the remote peer belongs are configured.

The number of the AS where the specified peer resides must be different from that of the local AS.

The IP address of the specified peer can be one of the following types:

- IPv6 address of an interface on a directly connected peer
- IP address of a routable loopback interface on the peer
- IPv6 address of a sub-interface on a directly connected peer
- Link-local address of an interface on a directly connected peer

If the specified peer IP address is a loopback interface address or a local-link address, you need to configure the local interface for the BGP4+ connection to ensure that the peer relationship is correctly established.

e. (Optional) Run **peer { ipv6-address | group-name } connect-interface interface-type interface-number [ ipv6-source-address ]**

The source interface and source address for establishing a TCP connection are specified.

 NOTE

When establishing multiple peer relationships between two routers through multiple links, you are advised to run the **peer connect-interface** command to specify the interface for establishing each BGP4+ connection.

f. Run **peer { ipv6-address | group-name } ebgp-max-hop [ hop-count ]**

The maximum number of hops is configured for establishing an EBGP connection.

A direct physical link must be available between EBGP peers. If such a link does not exist, the **peer ebgp-max-hop** command must be used to allow EBGP peers to establish a TCP connection over multiple hops.

 NOTE

If loopback interfaces are used to establish an EBGP peer relationship, the **peer ebgp-max-hop** command must be run with *hop-count* greater than or equal to 2; otherwise, the peer relationship cannot be established.

g. (Optional) Run **peer { ipv6-address | group-name } listen-only**

The local peer (group) is configured to accept connection requests, but not to send connection requests.

After this command is run, the existing peer relationship is interrupted. The local peer will wait for the connection request from the remote peer to reestablish a peer relationship. This command enables only one peer to send connection requests, preventing a connection request conflict.

 NOTE

This command can only be run on one of two peers. If this command is run on both peers, the connection between them cannot be reestablished.

- h. (Optional) Run **peer { ipv6-address | group-name } description**  
*description-text*

The description of the peer is configured.

You can simplify network management by configuring the descriptions of peers.

- i. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- j. Run **peer { ipv6-address | group-name } enable**

The IPv6 peer is enabled.

After configuring a BGP4+ peer in the BGP view, you also need to enable the peer in the IPv6 unicast address family view.

- k. (Optional) Run **peer { ipv6-address | peerGroupName } peer-as-check**

The device is configured not to advertise the routes learned from an EBGP peer to the peers in the same AS with the EBGP peer.

By default, after receiving a route from an EBGP peer (for example, in AS 200), the local device (for example, in AS 100) advertises the route to other EBGP peers in AS 200. After the **peer peer-as-check** command is run on the device, the device does not advertise the routes received from an EBGP peer to other EBGP peers in the same AS with this EBGP peer. This reduces BGP memory and CPU consumption and speeds up route convergence in case of route flapping.

- l. Run **commit**

The configuration is committed.

----End

## Configuring BGP4+ to Import Routes

BGP4+ can import routes from other protocols. When routes are imported from dynamic routing protocols, the process IDs of the routing protocols must be specified. Importing routes from other protocols can enrich the BGP4+ routing table. When importing IGP routes, BGP4+ can filter the routes by routing protocol.

## Context

BGP4+ cannot discover routes by itself. Instead, it imports routes discovered by other protocols such as OSPFv3 or static routes to the BGP4+ routing table. These imported routes are then transmitted within an AS or between ASs. When importing routes, BGP4+ can filter these routes by routing protocol.

BGP4+ can import routes in either Import or Network mode.

- Import mode: BGP4+ imports routes into its routing table by protocol type, such as RIP, OSPF, IS-IS, static routes, or direct routes.

- Network mode: BGP4+ imports a route with the specified prefix and mask. This mode is more precise than the Import mode.

## Procedure

- Import mode
  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp as-number**

The BGP view is displayed.
  - c. Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.
  - d. Run **import-route { direct | static | unr | { ospfv3| isis | ripng } process-id } [ [ med med ] | [ [ route-policy route-policy-name ] | [ route-filter route-filter-name ] ] ] \***

BGP4+ is configured to import routes from another protocol.  
To filter the routes imported from another protocol, you can specify the **route-policy route-policy-name** parameter.  
To filter the routes imported from another protocol, you can also specify the **route-filter route-filter-name** parameter.

### NOTE

To import IS-IS, OSPF, or RIP routes, the process ID of the corresponding routing protocol needs to be specified.

In a BAS device access scenario, to prevent BGP from importing the UNRs that cannot be advertised, run the **access no-advertise-unr import disable** command.

- Run **default-route imported**

BGP4+ is allowed to import default routes. BGP4+ can import default routes only when both the **default-route imported** and **import-route** commands have been run. Using only the **import-route** command is insufficient to import default routes, and using only the **default-route imported** command imports only the default routes that exist in the local routing table.
- Run **commit**

The configuration is committed.
- Network mode
  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp as-number**

The BGP view is displayed.
  - c. Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

- d. Run **network ipv6-address prefix-length [ route-policy route-policy-name | route-filter route-filter-name ]**

The device is configured to advertise local IPv6 routes.

If no mask or mask length is specified, the address is processed as a classful address.

The local routes to be advertised must exist in the local IP routing table.

To use a route-policy to filter the routes to be imported, specify the **route-policy route-policy-name** parameter.

To filter the routes imported from another protocol, you can also specify the **route-filter route-filter-name** parameter.

#### NOTE

The destination address and mask specified in the **network** command must be consistent with those in the corresponding entry in the local IP routing table. Otherwise, the specified route will not be advertised.

When using the **undo network** command to clear the existing configuration, specify the correct mask.

- e. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring basic BGP4+ functions, verify BGP4+ peer information.

## Prerequisites

Basic BGP4+ functions have been configured.

## Procedure

- Run the **display bgp ipv6 peer verbose** command to check information about all BGP4+ peers.
- Run the **display bgp ipv6 peer ipv6-address { log-info | verbose }** command to check information about a specified BGP4+ peer.
- Run the **display bgp ipv6 routing-table [ ipv6-address prefix-length ]** command to check information about BGP4+ routes.
- Run the **display bgp ipv6 routing-table as-path-filter { as-path-filter-number | as-path-filter-name }** command to check information about the routes that match the specified AS\_Path filter.

----End

### 1.1.10.1.4 Controlling Route Advertisement

BGP4+ can filter or apply routing policies to the routes to be advertised to a peer or peer group.

## Usage Scenario

BGP4+ advertises routes to peers based on the network plan to exchange routing information between devices. The routes can be filtered or processed based on a routing policy before being advertised to a peer or peer group.

## Pre-configuration Tasks

Before controlling route advertisement, [configure basic BGP4+ functions](#).

## Configuring BGP4+ Route Summarization

Configuring route summarization can reduce the size of a routing table on a peer.

## Context

On a large-scale BGP4+ network, configuring route summarization can reduce the number of advertised route prefixes and improve BGP4+ stability.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

### Step 4 Run one of the following commands to configure route summarization as needed.

- To advertise all summary routes and specific routes, run the **aggregate ipv6-address mask-length** command.
- To advertise only summary routes, run the **aggregate ipv6-address mask-length detail-suppressed** command.
- To advertise some of the specific routes, run the **aggregate ipv6-address mask-length suppress-policy route-policy-name** command.

Using the **peer route-policy** command can also advertise some of the specific routes.

- To generate a summary route used for loop detection, run the **aggregate ipv6-address mask-length as-set** command.

- To configure the attributes of summary routes, run the **aggregate ipv6-address mask-length attribute-policy route-policy-name** command.

You can also run the **peer route-policy** command to achieve the same effect.

If **as-set** is configured in the **aggregate** command, the **AS\_Path** attribute configured in the **apply as-path** command does not take effect.

- To generate summary routes based on some of the specific routes, run the **aggregate ipv6-address mask-length origin-policy route-policy-name** command.

Manual route summarization is valid for the routing entries that exist in the local BGP4+ routing table. For example, if the **aggregate 2001:db8::1 64** command is run to summarize routes, but no route with a mask length greater than 64 (for example, 2001:db8::1/128) exists in the BGP4+ routing table, BGP4+ does not advertise the summary route.

#### Step 5 Run **commit**

The configuration is committed.

----End

### Configuring BGP4+ to Advertise Default Routes to Peers or peer groups

A device sends a default route with the local address as the next hop address to the specified peer for load balancing, regardless of whether there are default routes in the local routing table. This greatly reduces the number of routes on the network.

### Context

Default routes can be used on networks that have the following characteristics:

- There are multiple EBGP peers, and each peer can receive full Internet routes.
- There are multiple Route Reflectors (RRs), and each RR can receive full Internet routes.

When load balancing is not performed on the network, a BGP4+ peer receives at most one copy of active full Internet routes. After load balancing is performed on the network, the number of active routes received by the BGP4+ peer doubles, which causes the number of routes on the network to sharply increase. In this case, you can configure the local device to advertise only default routes to its BGP4+ peer and use default routes for load balancing, which can greatly reduce the number of routes on the network.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bgp as-number**

The BGP view is displayed.

#### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

#### Step 4 Run **peer { ipv4-address | ipv6-address | group-name } default-route-advertise [ route-policy route-policy-name | route-filter route-filter-name ] [ conditional-route-match-all { ipv6-address1 ipv6-mask-length1 } &<1-4> | conditional-route-match-any { ipv6-address2 ipv6-mask-length2 } &<1-4> ]**

The device is configured to send the default route to the specified peer or peer group.

If **route-policy route-policy-name** or **route-filter route-filter-name** is set, the BGP device changes attributes of the default route accordingly.

- If **conditional-route-match-all { ipv6-address1 ipv6-mask-length1 } &<1-4>** is specified, the device advertises the default route only if all the routes matching the specified conditions exist in the local IPv6 routing table.
- If **conditional-route-match-any { ipv6-address2 ipv6-mask-length2 } &<1-4>** is specified, the device advertises the default route if any route matching a specified condition exists in the local IPv6 routing table.

 NOTE

After the **peer default-route-advertise** command is run on a device, the device advertises a default route (with its local address as the next-hop address) to the specified peer, regardless of whether there is a default route in the local routing table.

**Step 5 Run commit**

The configuration is committed.

----End

## Enabling a BGP4+ Device to Add the Community Attribute to Routes to Be Advertised

The community attribute is used to simplify route-policy management.

### Context

The community attribute is transmitted between BGP4+ peers, and its transmission is not restricted within ASs. With the community attribute, a group of routes can share the same route-policy. Before sending a route with the community attribute to peers, BGP4+ can change the original community attribute carried by the route.

### Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run bgp as-number**

The BGP view is displayed.

**Step 3 Run ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4 Run peer { ipv4-address | ipv6-address | group-name } route-policy route-policy-name export**

An export route-policy is applied to routes to be advertised to a specified peer or peer group.

 NOTE

- When configuring a BGP4+ community, use a route-policy to define a specific community attribute. Then, apply the route-policy to the routes to be advertised.
- For details about the route-policy configuration, see [Routing Policy Configuration](#).

**Step 5** Run either of the following commands to configure the device to advertise community attributes to a peer or peer group:

- To configure the device to advertise standard community attributes to a specified peer or peer group, run the **peer { ipv4-address | ipv6-address | group-name } advertise-community** command.
- To configure the device to advertise extended community attributes to a specified peer or peer group, run the **peer { ipv4-address | ipv6-address | group-name } advertise-ext-community** command.

**Step 6** Run **commit**

The configuration is committed.

----End

## Enabling a BGP4+ Device to Add the Large-Community Attribute to Routes to Be Advertised

The Large-Community attribute can be flexibly applied in route-policies.

### Context

The Large-Community attribute can represent a 2-byte or 4-byte Autonomous System Number (ASN), and has two 4-byte LocalData IDs. The administrator can therefore apply route-policies more flexibly. The Large-Community attribute extends and can be used together with a community attribute.

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Run **peer { ipv4-address | ipv6-address | group-name } route-policy route-policy-name export**

An export route-policy is applied to routes to be advertised to a specified peer or peer group.



- When configuring the BGP4+ Large-Community attribute, use a route-policy to define specific Large-Community values, and then apply the route-policy to the routes to be advertised.
- For details about the route-policy configuration, see [Routing Policy Configuration](#).

**Step 5** Run **peer { ipv4-address | ipv6-address | group-name } advertise-large-community**

The device is configured to advertise the Large-Community attribute to the specified peer or peer group.

**Step 6** Run **commit**

The configuration is committed.

----End

### (Optional) Setting the Interval at Which Update Messages Are Sent

When routes change, the router sends Update messages to notify its peers. If a route changes frequently, to prevent the router from sending Update messages upon every change, you can set an interval at which Update messages are sent as required.

### Context

Perform the following steps on a BGP4+ device:

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Run **peer { ipv4-address | ipv6-address | group-name } route-update-interval interval**

The interval at which Update messages are sent is set.

**Step 5** Run **commit**

The configuration is committed.

----End

### Verifying the Configuration

After controlling route advertisement, check information about filtered and advertised routes.

### Prerequisites

Configurations have been performed to control route advertisement.

### Procedure

- Run the **display bgp ipv6 network** command to check routing information advertised by BGP4+.

- Run the **display bgp routing-table cidr** command to check Classless InterDomain Routing (CIDR) information.
- Run the **display bgp ipv6 routing-table community [ aa:nn &<1-13> ] [ internet | no-advertise | no-export | no-export-subconfed ] \* [ whole-match ]** command to check information about the routes carrying the specified BGP4+ community attribute.
- Run the **display bgp ipv6 peer [ ipv6-address ] verbose** command to check information about BGP4+ peers.
- Run the **display bgp ipv6 peer ipv6-address log-info** command to check information about BGP4+ peers.

----End

### 1.1.10.1.5 Controlling BGP4+ Route Selection

The policy on BGP4+ route selection can be changed by configuring BGP4+ route attributes.

#### Usage Scenario

BGP4+ has many route attributes. These attributes are used to define the routing policy and describe the BGP4+ route prefix. Configuring these attributes can change the policy used by BGP4+ for route selection.

#### Pre-configuration Tasks

Before controlling BGP4+ route selection, [configure basic BGP4+ functions](#).

#### Setting a BGP4+ Preference

Setting the BGP4+ preference can control route selection between BGP routes and routes of another routing protocol.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bgp as-number**

The BGP view is displayed.

##### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

##### Step 4 Run **preference { external internal local | route-policy route-policy-name | route-filter route-filter-name }**

The BGP4+ preference is set.

BGP4+ has the following types of routes:

- EBGP routes, which are learned from peers in other ASs

- IBGP routes, which are learned from peers in the same AS
- Locally originated routes, which are summary route generated using the **aggregate** command

You can set different preferences for the three types of routes.

You can also apply routing policies to set preferences for the specified routes that meet the requirements. You can set default preferences for the routes that do not meet the requirements.

 **NOTE**

At present, the **peer route-policy** or **peer route-filter** command cannot be used to set the BGP4+ priority.

**Step 5 Run commit**

The configuration is committed.

----End

## Setting a PrefVal for BGP4+ Routes

After a PrefVal is set for BGP4+ routes, the route with the greatest value is preferred when multiple routes to the same destination exist in the BGP4+ routing table.

## Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run bgp *as-number***

The BGP view is displayed.

**Step 3 Run ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4 Run peer { *ipv4-address* | *ipv6-address* | *group-name* } **preferred-value** *preferredvalue***

The original PrefVal of a route learned from a peer defaults to 0.

After the **peer preferred-value** command is run, all the routes learned from a specified peer have the same PrefVal.

**Step 5 Run commit**

The configuration is committed.

----End

## Setting the Default Local\_Pref Attribute for the Local Device

After the Local\_Pref attribute is set for BGP4+ routes, the route with the greatest attribute value is preferred when multiple routes to the same destination exist in

the BGP4+ routing table. The preferred value takes precedence over the Local\_Pref attribute.

## Context

The Local\_Pref attribute is used to determine the optimal route for the traffic that leaves an AS. If a BGP device obtains multiple routes from different IBGP peers and these routes have different next hops to the same destination, the BGP device will select the route with the greatest Local\_Pref value.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

### Step 4 Run **default local-preference local-preference**

### Step 5 Run **commit**

The configuration is committed.

----End

## Setting the MED Attribute

The MED attribute is equal to the metric used in IGP. After the MED attribute is set for routes, an EBGP peer can select a route with the smallest MED value for the traffic that enters an AS.

## Context

The MED serves as the metric used by an IGP. It is used to determine the optimal route when traffic enters an AS. When a BGP4+ router obtains multiple routes to the same destination address but with different next hops through EBGP peers, the route with the smallest MED value is selected as the optimal route.

## Procedure

- Set the default MED value on the local device.

- Run **system-view**

The system view is displayed.

- Run **bgp as-number**

The BGP view is displayed.

- Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run **default med med**  
The default MED value is set.
  - e. Run **commit**  
The configuration is committed.
- Compare the MED values of the routes from different ASs.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **bgp as-number**  
The BGP view is displayed.
    - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
    - d. Run **compare-different-as-med**
    - e. Run **commit**  
The configuration is committed.
  - Configure the method used by BGP4+ when there is no MED attribute in the route attributes.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **bgp as-number**  
The BGP view is displayed.
    - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
    - d. Run **bestroute med-none-as-maximum**  
The maximum MED value is used for a route when there is no MED attribute in the route attributes.  
If this command is not run, BGP4+ uses 0 as the MED value for a route when there is no MED attribute in the route attributes.
    - e. Run **commit**  
The configuration is committed.

----End

## Setting the Next\_Hop Attribute

BGP4+ route selection can be flexibly controlled by setting the Next\_Hop attribute.

## Context

The Next\_Hop attribute of BGP4+ is different from that of an IGP because it is not necessarily the IPv6 address of a neighboring router.

## Procedure

- Change the next-hop address when advertising a route to an IBGP peer.

Perform the following steps on the IBGP router:

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run **peer { ipv6-address | group-name } next-hop-local**

The device is configured to use its own IP address as the next-hop address of the routes when advertising them.

To ensure that an IBGP peer can find the correct next hop, you can configure the local device to use its own address as the next-hop address of each route when the local device advertises routes to the IBGP peer.

### NOTE

If BGP load balancing is configured, the local router changes the next-hop address of a route to its own IP address when advertising the route to an IBGP peer, regardless of whether the **peer next-hop-local** command is run.

- e. Run **commit**

The configuration is committed.

- Configure a device to retain the original Next\_Hop of imported IGP routes when the device advertises the routes to an IBGP peer.

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run **peer { peerIpv6Addr | peerGroupName } next-hop-invariable [ include-static-route | include-unicast-route ] \***

The device is configured not to change the next hop address of an imported IGP route when advertising the route.

If the **peer next-hop-invariable include-static-route** command is run, the BGP speaker retains the original next hop address of an imported public network static route when advertising the route to an IBGP peer under the condition that the original next hop address is valid; if the original next hop address of the static route is invalid, the public network static route recurses to a VPN route, or the public network static route is imported from a VPN instance, the BGP speaker uses its interface address as the next hop of the route.

If the **peer next-hop-invariable include-unicast-route** command is run, the BGP speaker does not change the next hop address when advertising to an EBGP peer the unicast routes learned from another peer.

e. Run **commit**

The configuration is committed.

- Prevent an ASBR from changing the next hop address when advertising routes to an EBGP peer.

Perform the following steps on a PE:

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv6-family vpng6 [ unicast ]**

The BGP-VPNG6 unicast address family view is displayed.

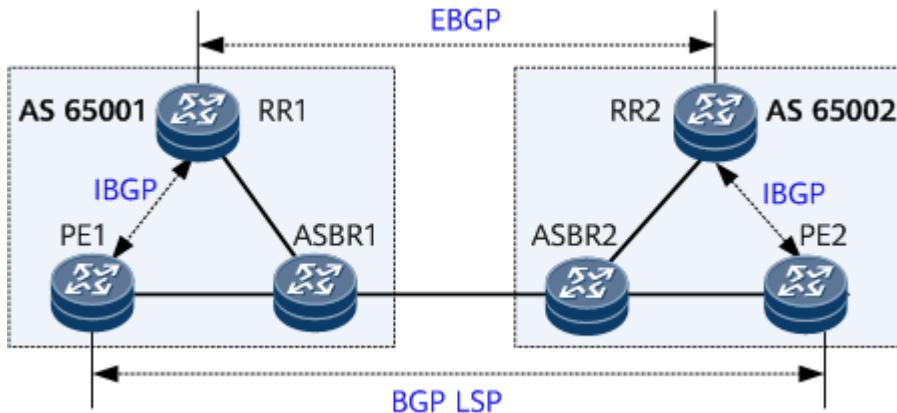
- d. Run the **peer { ipv6-address | group-name } next-hop-invariable** command to configure the device not to change the next hop when advertising routes to the specified EBGP peer. Alternatively, run the **peer { ipv6-address | group-name } next-hop-invariable [ include-static-route | include-unicast-route ] \* command.**

If the **peer next-hop-invariable include-static-route** command is run, the BGP speaker retains the original next hop address of an imported public network static route when advertising the route to an IBGP peer under the condition that the original next hop address is valid; if the original next hop address of the static route is invalid, the public network static route recurses to a VPN route, or the public network static route is imported from a VPN instance, the BGP speaker uses its interface address as the next hop of the route.

If the **peer next-hop-invariable include-unicast-route** command is run, the BGP speaker does not change the next hop address when advertising to an EBGP peer the unicast routes learned from another peer.

On the network shown in [Figure 1-452](#), a BGP LSP is established between PE1 and PE2. VPNG6 routes are exchanged between PE1 and RR1, between RR1 and RR2, and between RR2 and PE2 through BGP VPNG6 peer relationships.

Figure 1-452 Inter-AS VPN Option C networking with RRs deployed



Assume that PE1 needs to advertise a VPNv6 route to PE2. The route will be advertised through the following process:

- i. PE1 advertises the route to RR1, with the route next hop being PE1.
- ii. Upon receipt, RR1 changes the route next hop to itself and then advertises the route to RR2 through the EBGP peer relationship.
- iii. RR2 advertises the received route to its IBGP peer PE2. By default, the router changes the next hop of a labeled route received from an EBGP peer before advertising the route to an IBGP peer. Therefore, RR2 changes the next hop of the route to itself before advertising the route to PE2.

The next hop of the VPNv6 route received by PE2 is RR2. However, the destination of the BGP LSP is PE1. As a result, the VPNv6 route on PE2 cannot recurse to the BGP LSP, causing traffic interruption.

To solve the preceding problem, run the **peer next-hop-invariable** command on RR1 to ensure that RR1 does not change the next hop when advertising routes to RR2. In addition, run the **peer next-hop-invariable** command on RR2 to ensure that RR2 does not change the next hop when advertising routes to PE2. In this way, the next hop of the VPNv6 route received by PE2 is PE1, and the route can recurse to the BGP LSP.

Similar to the preceding process, if PE2 also attempts to advertise a VPNv6 route to PE1, run the **peer next-hop-invariable** command on both RR2 and RR1 so that RR2 does not change the next hop before advertising the route to RR1 and RR1 does not change the next hop before advertising the route to PE1. In this way, the VPNv6 route received by PE1 is PE2, and the route can recurse to the BGP LSP.

e. Run **commit**

The configuration is committed.

- Configure route-policy-based next hop recursion.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run **nexthop recursive-lookup { route-policy route-policy-name | route-filter route-filter-name }**  
Route-policy-based next hop recursion is configured.  
Next-hop recursion based on a specified route-policy can control the recursive next hop based on specific conditions. If a route fails to match the specified route-policy, the route recursion fails.
  - e. Run **commit**  
The configuration is committed.
- Prevent the device from changing the next hop address of a route to ensure that traffic is transmitted along the optimal route when the device advertises the route to a peer in the following scenarios:
    - The route is learned from a directly connected peer and is to be advertised to a directly connected EBGP peer, the original next hop of the route resides on the same network segment as the local interface that is used to establish the BGP4+ peer relationship with the EBGP peer, and directly connected interfaces are broadcast interfaces.
    - The route is locally imported and is to be advertised to a directly connected IBGP or EBGP peer, the next hop to which the route recurses resides on the same network segment as the local interface that is used to establish the BGP4+ peer relationship with the IBGP or EBGP peer, and directly connected interfaces are broadcast interfaces.
- a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run **nexthop third-party**  
The device is prevented from changing the next hop address of a route when the device advertises the route to a peer in the specified scenarios.
  - e. Run **commit**  
The configuration is committed.

----End

## Setting the AS\_Path Attribute

The AS\_Path attribute is used to prevent routing loops and control route selection.

## Procedure

- Set the AS\_Path attribute in the IPv6 address family view.  
Perform the following steps on a BGP4+ device:

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

d. Run one of the following commands to configure the AS\_Path attribute.

- To enable the device to accept the routes that contain the local AS number if the number of repetitions in the route is within the configured limit, run the **peer { ipv6-address | group-name } allow-as-loop [ number ]** command.
- To exclude the AS\_Path attribute from being used as a route selection rule, run the **bestroute as-path-ignore** command.
- To configure the AS\_Path attribute to carry only public network AS numbers, run the **peer { ipv6-address | group-name } public-as-only [ force [ replace ] [ include-peer-as ] | limited [ replace ] [ include-peer-as ] ] or peer { ipv6-address | group-name } public-as-only import [ force ]** command.

The commands in Step 4 are optional and can be used in random order.

e. Run **commit**

The configuration is committed.

• Configure a fake AS number.

Perform the following steps on a BGP4+ device:

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **peer { ipv6-address | group-name } fake-as fake-as-value [ dual-as ] [ prepend-global-as ] [ prepend-fake-as ]**

A fake AS number is configured.

You can run this command to hide the actual AS number of the local device. EBGP peers in other ASs can view only the fake AS number. That is, EBGP peers in other ASs must use the fake AS number when specifying the AS number of the local peer.

 NOTE

This command is applicable only to EBGP peers.

d. Run **commit**

The configuration is committed.

• Substitute the AS numbers in the AS\_Path attribute.

In a BGP/MPLS IP VPN scenario, if the ASs to which two VPN sites belong use private AS numbers, the AS numbers of the two VPN sites may be the same. If a CE in a VPN site sends a VPN route to the connected PE using EBGP and the PE then sends the route to the remote CE, the remote CE will discard the route because the AS number carried by the route is the same as the local AS number. As a result, different sites of the same VPN cannot communicate with each other. In this case, you need to run the **peer substitute-as** command on the PE to enable AS number substitution. That is, the PE replaces the AS number of the VPN site where the CE resides in the received VPN route with the local AS number. This prevents the remote CE from discarding routes due to duplicate AS numbers.

In a BGP public network scenario, when two devices with the same AS number learn a BGP route from each other through the same EBGP peer, the route may be discarded because the AS\_Path attribute contains duplicate AS numbers. To prevent this problem, run the **peer substitute-as** command on the EBGP peer shared by the two devices to enable AS number substitution.

#### NOTICE

Exercise caution when running the **peer substitute-as** command because improper use of the command may cause routing loops.

- a. Run the **system-view** command to enter the system view.
  - b. Run the **bgp as-number** command to enter the BGP view.
  - c. Run the **ipv6-family vpn-instance vpn-instance-name** command to enter the BGP-VPN instance IPv6 address family view.
  - d. Run the **peer { ipv6-address | group-name } substitute-as** command to enable AS number substitution on the device.
  - e. Run the **commit** command to commit the configuration.
- Enable the device to check or disable the device from checking the first AS number in the AS\_Path attribute contained in the update messages received from a specified EBGP peer or peer group.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **bgp as-number**  
The BGP view is displayed.
    - c. (Optional) Run **ipv6-family vpn-instance vpn-instance-name**  
The BGP-VPN instance IPv6 address family view is displayed.
    - d. Run **peer { ipv6-address | group-name } check-first-as { enable | disable }**  
The device is enabled to check or disabled from checking the first AS number in the AS\_Path attribute contained in the update messages received from a specified EBGP peer or peer group.  
If the **peer check-first-as enable** command is run, the device checks whether the first AS number in the AS\_Path attribute contained in the

update messages received from the specified EBGP peer or peer group is the number of the AS where the EBGP peer or peer group resides. If the two AS numbers are different, the local device discards the update messages. If the **peer check-first-as disable** command is run, the device accepts all update messages received from the specified EBGP peer or peer group, regardless whether the two AS numbers are the same. If the **undo peer check-first-as disable** command is run, the default configuration takes effect.

The check function can be configured for a specified EBGP peer, peer group, or for BGP as a whole. If the function is not configured for a specified EBGP peer, the device checks whether the function is configured for the related peer group; if the function is not configured for the peer group, the device checks whether the function is configured in the BGP view.

e. Run **commit**

The configuration is committed.

After the configuration is complete, if you want to check the received routes again, run the **refresh bgp** command.

----End

## Configuring AIGP Attributes for Routes

The Accumulated Interior Gateway Protocol (AIGP) attribute allows devices in an AIGP domain to use the optimal routes to forward data.

### Context

Generally, a set of ASs managed by the same administrative department is called an AIGP administrative domain, or AIGP domain for short.

Routing protocols that run within a single administrative domain, such as IGPs, assign a metric to each link and select the path with the smallest metric. BGP, designed to provide routing over a large number of independent administrative domains, does not select paths based on metrics. If a single administrative domain (AIGP domain) consists of several BGP networks, it is desirable for BGP to select paths based on metrics, just as an IGP does.

After the AIGP attribute is configured in an AIGP domain, BGP selects optimal routes based on route metrics, just as an IGP does. This ensures that all devices in the AIGP domain forward data along the optimal paths.

### Procedure

**Step 1** Enable AIGP capability for a BGP peer or peer group.

1. Run **system-view**

The system view is displayed.

2. Run **bgp as-number**

The BGP view is displayed.

3. Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

4. Run **peer { group-name | ipv6-address } aigp**

The AIGP capability is enabled for a BGP peer or peer group.

BGP allows you to enable the AIGP capability for either a BGP peer or a BGP peer group. The configuration on a peer takes precedence over that on a peer group. If a BGP peer without the AIGP capability joins a BGP peer group that has the AIGP capability, the BGP peer inherits the AIGP capability of the BGP peer group. After a BGP peer inherits the AIGP capability of a BGP peer group, you can run the **undo peer aigp** command to delete the AIGP configuration from the BGP peer.

5. Run **commit**

The configuration is committed.

**Step 2** (Optional) Allow IPv6 public network routes to participate in route selection using the AIGP attribute of the corresponding BGP LSP.

1. Run the **system-view** command to enter the system view.
2. Run the **bgp as-number** command to enter the BGP view.
3. Run the **ipv6-family unicast** command to enter the BGP-IPv6 unicast address family view.
4. Run the **bestroute nexthop-resolved aigp** command to allow IPv6 public network routes to participate in route selection using the AIGP attribute of the corresponding BGP LSP.
5. Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

After configuring BGP4+ route attributes, verify information about the route attributes.

## Prerequisites

BGP4+ route attributes have been configured.

## Procedure

- Run the **display bgp ipv6 routing-table different-origin-as** command to check the routes with different origin ASs.
- Run the **display bgp ipv6 routing-table regular-expression as-regular-expression** command to check the routes that match the AS regular expression.
- Run the **display bgp ipv6 routing-table community [ aa:nn &<1-33> ] [ internet | no-advertise | no-export | no-export-subconfed ] \* [ whole-match ]** command to check the routes carrying the specified BGP4+ community attribute.
- Run the **display bgp ipv6 routing-table community-filter { { community-filter-name | basic-community-filter-number } [ whole-match ] | advanced-**

*community-filter-number}* } command to check the routes that match the specified BGP4+ community filter.

----End

### 1.1.10.1.6 Configuring BGP4+ Routing Policies

BGP4+ routing policies can be configured to flexibly control the sending and receiving of routes.

#### Usage Scenario

Routing policies can set or re-set BGP4+ route attributes using some predefined conditions, which provides a flexible and effective method to control BGP4+ route selection. The sending and receiving of routes can be flexibly controlled by applying BGP4+ routing policies.

Based on the import (or export) routing policy specified by the peer, the associated import (or export) routing conditions (**if-match** clauses) can be configured to filter routes, and **apply** clauses can be configured to set or modify route attributes. The routes that match the routing policy will be received (or sent).

#### Pre-configuration Tasks

Before configuring BGP4+ routing policies, complete the following tasks:

- Configure IP addresses for interfaces to ensure IP connectivity between neighboring nodes.
- [Configure basic BGP4+ Functions](#).

#### Configuring BGP4+ Filters

BGP4+ filters can be used in routing policies or when you want to check BGP4+ running status as required.

#### Context

BGP4+ provides the following types of filters, which can be used to query BGP4+ running status or used in routing policies:

- [\*\*AS\\_Path filter\*\*](#)  
The AS\_Path filter filters BGP4+ routes by the AS\_Path attribute and filters out unqualified routes. Multiple rules (permit or deny) can be specified in a filter.
- [\*\*Community filter\*\*](#)  
The community filter consists of multiple community attribute lists. There are two types of community attribute lists: standard community access lists and extended community access lists.
- [\*\*IPv6 address prefix list\*\*](#)  
Before configuring a conditional BGP4+ route advertisement policy, you need to create an IPv6 address prefix list.

## Procedure

- Configure the AS\_Path filter.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **ip as-path-filter { as-path-filter-number | as-path-filter-name } [ index index-number ] matchMode regular-expression**

The AS\_Path filter is configured.

After the **peer as-path-filter** command is used to apply a routing policy to BGP4+ routes, the AS\_Path filter filters out unqualified routes.

The AS\_Path filter uses the regular expression to define matching rules. A regular expression consists of the following parts:

- Metacharacter: defines matching rules.
- General character: defines matching objects.

**Table 1-158** Description of metacharacters

| Special Character | Function                                     | Example                                                                                                                                                                                                                                                                                                                                                                             |
|-------------------|----------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| .                 | Matches any single character.                | .* matches any string in an AS_Path and is used to match any route.                                                                                                                                                                                                                                                                                                                 |
| ^                 | Indicates the beginning of a matched string. | ^65 matches strings beginning with 65. <ul style="list-style-type: none"><li>● Examples of matched strings: 65, 651, 6501, and 65001</li><li>● Examples of unmatched strings: 165, 1650, 6650, and 60065</li></ul>                                                                                                                                                                  |
| \$                | Indicates the end of a matched string.       | 65\$ matches strings ending with 65. <ul style="list-style-type: none"><li>● Examples of matched strings: 65, 165, 1065, 10065, and 60065</li><li>● Examples of unmatched strings: 651, 1650, 6650, 60650, and 65001</li></ul><br>^65\$ matches AS_Path 65 only.<br><b>NOTE</b><br>^\$ matches an empty string (empty AS_Path) and is usually used to match routes in the local AS. |

| Special Character | Function                                                                                                                                                                                                                                                                                                                                     | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -                 | <p>Matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning of a regular expression with the same function as the caret sign (^) or at the end of a regular expression with the same function as the dollar sign (\$).</p> | <ul style="list-style-type: none"><li>• ^65001_ matches the AS_Paths that begin with 65001 followed by a sign. Specifically, ^65001_ matches AS_Paths with 65001 as the leftmost AS number (the number of the last AS through which a route passes) and the routes sent by peers in AS 65001.</li><li>• _65001_ matches the strings (AS_Paths) that contain 65001, which is used to match the routes that pass through AS 65001.</li><li>• _65001\$ matches the AS_Paths that end with a sign followed by 65001. Specifically, _65001\$ matches AS_Paths with 65001 as the rightmost AS number (the number of the first AS through which a route passes), which is used to match the routes that originate in AS 65001.</li></ul> |

| Special Character | Function                                                                                                                | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------------------|-------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| \                 | <p>Defines an escape character, which is used to mark the next character (common or special) as a common character.</p> | <p>An AS_Confed_Sequence contains parentheses. The parentheses in regular expressions provide special functions. To match such special characters by removing their special meanings, you can use the backslash (\). For example:</p> <ul style="list-style-type: none"><li>• \(65002_ matches the AS_Confed_Sequences that begin with (65002 followed by a sign. Specifically, \(65002_ matches AS_Confed_Sequences with 65002 as the leftmost AS number (the number of the last AS through which a route passes) and the routes sent by peers in AS 65002.</li><li>• \(.*_65003_*\) matches the AS_Confed_Sequence that contains AS number 65003 and the routes that pass through AS 65003 in a confederation.</li><li>• _65004\)) matches a string that ends with 65004 and with a sign before 65004. That is, the most significant AS number (start AS) of AS_Confed_Sequence is 65004. This string can also be used to match the routes originating in AS 65004 in a confederation and the routes directly advertised by AS 65004 in the confederation. _65004\)) provides the same function as 65004\)).</li></ul> <p>Similarly, backslashes (\) can be used to remove the special meanings of the left bracket ([) and right bracket (]) used</p> |

| Special Character | Function                                                                              | Example                                                                                                                                                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                   |                                                                                       | in an AS_Confederation_Set and the left brace {} and right brace {} used in an AS_Set.                                                                                                                                                                                              |
| *                 | Matches the strings in which the preceding character occurs zero or multiple times.   | <p>65* matches the AS_Paths that begin with 6 and contain zero or multiple 5s.</p> <ul style="list-style-type: none"> <li>• Examples of matched strings: 6, 65, 655, 6559, 65259, and 65529</li> <li>• Examples of unmatched strings: 5, 56, 556, 5669, 55269, and 56259</li> </ul> |
| +                 | Matches the strings in which the preceding character occurs one or more times.        | <p>65+ matches the AS_Paths that begin with 6 and contain one or multiple 5s.</p> <ul style="list-style-type: none"> <li>• Examples of matched strings: 65, 655, 6559, 65259, and 65529</li> <li>• Examples of unmatched strings: 56, 556, 5669, 55269, and 56259</li> </ul>        |
| ?                 | Matches the strings in which the preceding character occurs zero or one time.         | <p>65? matches the AS_Paths that begin with 6 and contain zero or one 5.</p> <ul style="list-style-type: none"> <li>• Examples of matched strings: 6 and 65</li> <li>• Examples of unmatched strings: 655, 6559, and 65529</li> </ul>                                               |
| ()                | Defines a subexpression, which can be empty. The parentheses can be empty in between. | 100(200)+ matches 100200, 100200200, and so on.                                                                                                                                                                                                                                     |
| x y               | Matches x or y.                                                                       | 100 65002 65003 matches 100, 65002, or 65003.                                                                                                                                                                                                                                       |
| [xyz]             | Matches any character in the regular expression.                                      | [896] matches 8, 9, or 6.                                                                                                                                                                                                                                                           |
| [^xyz]            | Matches any character that is not contained in the regular expression.                | [^896] matches any character, except 8, 9, and 6.                                                                                                                                                                                                                                   |

| Special Character | Function                                          | Example                                                                                                                                                                                                                                                                                       |
|-------------------|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [a-z]             | Matches any character within the specified range. | [2-4] matches any of 2, 3, and 4; [0-9] matches any digits from 0 to 9.<br><b>NOTE</b><br>The value in the square brackets ([]) must be a digit from 0 to 9. For example, to match a number ranging from 735 to 907, use the regular expression of (73[5-9] 7[4-9][0-9] 8[0-9][0-9] 90[0-7]). |
| [^a-z]            | Matches any character beyond the specified range. | [^2-4] matches AS_Paths without 2, 3, and 4, and [^0-9] matches AS_Paths without digits from 0 to 9.                                                                                                                                                                                          |

For example, ^10 indicates that only the AS\_Path attribute with the first value being 10 is matched. A circumflex (^) indicates that the beginning of a character string is matched.

Multiple rules, permit or deny, can be specified in a filter. The relationship between these rules is "OR". This means that if a route meets one of the matching rules, it matches the AS\_Path filter.

#### NOTE

For details on the regular expression, see **Regular Expression** in "Command Display Mode" of "How to Use the CLI" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - Basic Configuration*.

#### c. Run **commit**

The configuration is committed.

#### • Configure the community filter.

Community filters are classified into two types: standard community filters and advanced community filters. Advanced community filters support regular expressions and are more flexible than standard community filters.

##### a. Run **system-view**

The system view is displayed.

##### b. Configure a community filter.

- To configure a standard community filter, run the **ip community-filter basic basCfName [ index index-val ] matchMode [ cmntyStr | cmntyNum | internet [ strict-match ] | no-advertise | no-export | no-export-subconfed ] &<1-20>** or **ip community-filter cfIndex [ index index-val ] matchMode [ cmntyStr | cmntyNum | internet [ strict-match ] | no-advertise | no-export | no-export-subconfed ] &<1-20>** command.

- To configure an advanced community filter, run the **ip community-filter { advanced comm-filter-name | adv-comm-filter-num } [ index index-number ] matchMode regular-expression** command.
- c. Run **commit**

The configuration is committed.
- Configure a Large-community filter.

There are two types of Large-community filters: basic Large-community filters and advanced Large-community filters. Advanced Large-community filters support regular expressions and are more flexible than basic Large-community filters.

  - a. Run **system-view**

The system view is displayed.
  - b. Configure a Large-Community filter.
    - To configure a basic Large-Community filter, run the **ip large-community-filter basic large-comm-filter-name [ index index-number ] { permit | deny } { cmntyStr } &<1-16>** command.
    - To configure an advanced Large-Community filter, run the **ip large-community-filter advanced large-comm-filter-name [ index index-number ] { permit | deny } regular-expression** command.
  - c. Run **commit**

The configuration is committed.
- Configure an extended community filter.
  - a. Run **system-view**

The system view is displayed.
  - b. Perform either of the following operations as required to configure an extended community filter.

To configure a VPN-Target extended community filter:

  - To configure a basic VPN-Target extended community filter, run the **ip extcommunity-filter { basic-extcomm-filter-num | basic basic-extcomm-filter-name }[ index index-number ] { deny | permit } { rt extCmntyStr } &<1-16>** command.
  - To configure an advanced VPN-Target extended community filter, run the **ip extcommunity-filter { advanced-extcomm-filter-num | advanced advanced-extcomm-filter-name } [ index index-number ] { deny | permit } regular-expression** command.

To configure an SoO extended community filter:

  - To configure a basic SoO extended community filter, run the **ip extcommunity-list soo basic basic-extcomm-filter-name [ index index-number ] { permit | deny } { site-of-origin } &<1-16>** command.

- To configure an advanced SoO extended community filter, run the **ip extcommunity-list soo advanced advanced-extcomm-filter-name [ index index-number ] { permit | deny } regular-expression** command.

Multiple entries can be defined in an extended community filter. The relationship between the entries is "OR". This means that if a route matches one of the rules, the route matches the filter.

c. Run **commit**

The configuration is committed.

- Configure an IPv6 address prefix list.

Before configuring a conditional BGP4+ route advertisement policy, you need to create an IPv6 address prefix list.

a. Run **system-view**

The system view is displayed.

b. Run **filter-list ipv6-prefix name**

An IPv6 address prefix list is created, and its view is displayed.

c. Run **prefix address maskLen**

An IPv6 address and mask are configured for the IPv6 address prefix list.

d. Run **commit**

The configuration is committed.

----End

## Configuring a Route-Policy

A route-policy is used to match routes or route attributes, and to change route attributes when the matching rules are met.

### Context

A route-policy is used to match routes or route attributes, and to change route attributes when the matching rules are met.

A route-policy consists of multiple nodes, and each node can comprise the following clauses:

- **if-match clause**

The clauses define the matching rules of a route-policy. The matching objects are route attributes.

- **apply clause**

The clauses specify actions. Configuration commands are run after a route meets the matching rules specified by the **if-match** clauses. The **apply** clauses can change certain route attributes.



This section describes only the BGP4+ route-policy. For detailed information about route-policy configurations, see "Routing Policy Configuration."

## Procedure

### Step 1 Create a route-policy.

1. Run **system-view**

The system view is displayed.

2. Run **route-policy route-policy-name matchMode node node**

A route-policy is created, and the view of the route-policy is displayed.

3. Run **commit**

The configuration is committed.

### Step 2 Configure **if-match** clauses.

1. Run the following command to configure the **if-match** clause for the current node in the route-policy as required:

- To match BGP4+ routes against the AS\_Path attribute, run the **if-match as-path-filter { as-path-filter-number | as-path-filter-name } &<1-16>** command.
- To match BGP4+ routes against the community attribute:
  - **if-match community-filter { basic-comm-filter-num [ whole-match ] | adv-comm-filter-num [ sort-match ] }\* &<1-16>**
  - **if-match community-filter comm-filter-name [ whole-match | sort-match ]**
- To match BGP4+ routes against the Large-community attribute, run the **if-match large-community-filter large-comm-filter-name [ whole-match ]** command.
- To match BGP4+ routes against the VPN target extended community attribute, run the **if-match extcommunity-filter { { basic-extcomm-filter-num [ matches-all | whole-match ] | adv-extcomm-filter-num } &<1-16> | extcomm-filter-name [ matches-all | whole-match ] }** command.
- To match BGP4+ routes against the SoO extended community attribute, run the **if-match extcommunity-list soo extcomm-filter-name** command.

The commands can be executed in any sequence. A node may have multiple or no **if-match** clauses.

#### NOTE

- The relationship between the **if-match** clauses for a node of a route-policy is "AND". A route must meet all the matching rules before the action defined by the **apply** clause is performed.
- If no **if-match** clause is specified, all the routes are matched.

2. Run **commit**

The configuration is committed.

### Step 3 Configure the **apply** clause.

1. Run **system-view**

The system view is displayed.

2. Run **route-policy route-policy-name matchMode node node**

The route-policy view is displayed.

3. Run the following command as needed to configure the **apply** clause for the current node in the route-policy:

- To replace the AS numbers in the AS\_Path attribute of BGP4+ routes or add the specified AS number to the AS\_Path attribute, run the **apply as-path { as-number-plain | as-number-dot } &<1-128> { additive | overwrite | delete }** or **apply as-path asValues { additive | overwrite | delete }** command.
- To delete the specified community attribute of BGP4+ routes, run the **apply comm-filter comm-filter-number delete** command.

#### NOTE

The **apply comm-filter delete** command deletes the specified community attribute from routes. Running the **ip community-filter** command specifies only one community attribute each time. To delete more than one community attribute, run the **ip community-filter** command multiple times. If multiple community attributes are specified in one filter, none of them can be deleted. For example, see the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Command Reference*.

- To delete community attributes of BGP4+ routes, run the **apply community none** command.
- To set a community attribute for BGP4+ routes, run the **apply community { { community-number | aa:nn } &<1-32> | internet | no-advertise | no-export | no-export-subconfed }\* [ additive ]** command.

#### NOTE

Before running the **apply community community-list community-list-name** command, you need to run the **ip community-list** command to configure a BGP community list and run the **community** command to configure community attributes for the list.

- To delete the Large-Community attribute of BGP4+ routes, run the **apply large-community none** command.
- To set the Large-Community attribute of BGP4+ routes, run the **apply large-community { aa:bb:cc } &<1-16> { additive | overwrite | delete }** or **apply large-community-list large-community-list-name { additive | overwrite | delete }** command.
- To set the VPN-Target extended community attribute for BGP4+ routes, run the **apply extcommunity { rt { as-number:nn | ipv4-address:nn } } &<1-16> [ additive ]** command.
- To set the SoO extended community attribute for BGP4+ routes, run the **apply extcommunity soo { site-of-origin } &<1-16> additive** command.
- To set the bandwidth extended community attribute for BGP4+ routes, run the **apply extcommunity bandwidth { extCmntyString | none }** or **apply extcommunity bandwidth aggregate [ limit bandwidth-value ]** command.
- To set a MED value for BGP4+ routes, run the **apply cost { [ apply-type ] cost | inherit | none }** command.

- To set the MED value of BGP4+ routes as the IGP cost of the next hop, run the **apply cost-type internal** command.

 NOTE

If both the **apply cost** and **apply cost-type** commands are run, only the **apply cost** command takes effect.

- To set the VPN-Target extended community attribute for BGP4+ routes, run the **apply extcommunity { rt { as-number:nn | ipv4-address:nn } } &<1-16> [ additive ]** command.
- To set the local preference for BGP4+ routes, run the **apply local-preference [ + | - ] preference** command.
- To set the Origin attribute for BGP4+ routes, run the **apply origin { igp | egp { as-number-plain | as-number-dot } | incomplete }** command.
- To set the preferred value for BGP4+ routes, run the **apply preferred-value preferred-value** command.
- To set dampening parameters for EBGP routes, run the **apply dampening half-life-reach reuse suppress ceiling** command.

The commands in Step 3 can be configured in random order.

4. Run **commit**

The configuration is committed.

----End

## Applying a Policy to BGP4+ Route Advertisement

If BGP4+ is configured to filter received routes, only the routes that meet the matching rules are added to the local BGP4+ routing table and advertised to BGP4+ peers.

## Context

BGP4+ can apply a routing policy to all the routes to be advertised or only to the routes to be advertised to a specified peer or peer group. If multiple filter policies are configured, BGP advertises only routes that match all the filter policies.

## Procedure

- Configure BGP to filter all the routes to be advertised.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run **filter-policy { acl6-number | acl6-name acl6-name | ipv6-prefix ipv6-prefix-name } export [ direct | isis process-id | ospfv3 process-id | ripng process-id | static | unr ]**

The device is configured to filter the routes to be advertised.

After BGP4+ filters the routes imported using the **import-route** command, only those routes that match the filtering conditions are added to the local BGP4+ routing table and advertised to BGP4+ peers.

If *protocol* is specified, only routes of the specified protocol are filtered. If *protocol* is not specified, all BGP routes to be advertised are filtered, including the routes imported using the **import-route** or **network** command.

e. Run **commit**

The configuration is committed.

- Apply a routing policy to the routes to be advertised to a certain peer (group).
  - a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run the following command as needed to configure BGP4+ to use a specified filter to filter the routes to be advertised to a peer.

■ Based on a basic ACL:

- 1) Run **peer { ipv4-address | ipv6-address | group-name } filter-policy { acl6-number | acl6-name acl6-name } export**

The routes to be advertised to the specified peer or peer group are filtered.

- 2) Run **quit**

Return to the BGP view.

- 3) Run **quit**

Return to the system view.

- 4) Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

- 5) Run **rule [ rule-id ] [ name rule-name ] { deny | permit }\***

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.

- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.  
Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- To use an AS\_Path filter to filter routes, run the **peer { ipv4-address | ipv6-address | group-name } as-path-filter { number | name } export** command.
- To use an IP prefix list to filter routes, run the **peer { ipv4-address | ipv6-address | group-name } ipv6-prefix ipv6-prefix-name export** command.
- To use a route-policy for route filtering, run the **peer { ipv4-address | ipv6-address | group-name } route-policy route-policy-name export** command.
- To use an IPv6 address list to filter routes, run the **peer { peerIpv4Addr | peerIpv6Addr | groupName } advertise dependent-filter dependent-filter-list outDependType [ condition-filter condition-filter-list | condition-ip-filter ip-prefix-name ]** command.

#### NOTE

If BGP route status changes after a filtering policy that is based on an IPv6 address list is enabled, the routes are re-advertised based on the conditional advertisement policy 10 seconds later by default. To set the delay in advertising the routes, run the **timer dependent-advertise-delay delay-time** command.

The export routing policy applied to a peer in a peer group can be different from that applied to the peer group. Specifically, a unique policy can be used to filter the routes to be advertised to each peer in the peer group.

e. Run **commit**

The configuration is committed.

----End

## Configuring a Policy for Receiving BGP4+ Routes

BGP4+ filters received routes using a policy. Only the routes that match the policy can be added to a routing table.

### Context

BGP4+ can apply a routing policy to all received routes or only routes received from a specific peer (group). If multiple filter policies are configured, BGP accepts only routes that match all the filter policies.

If a BGP4+ router is maliciously attacked or network configuration errors occur, the device will receive a large number of routes from its peers. As a result, a large number of resources are consumed on the device. To prevent this issue, the administrator must limit the resources to be consumed during device operation based on the network planning and router capacity. BGP4+ provides peer-specific route control to limit the number of routes sent from a peer or peer group.

### Procedure

- Configure BGP4+ to filter all received routes.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run **filter-policy { acl6-number | acl6-name acl6-name | ipv6-prefix ipv6-prefix-name } import**  
A policy is configured to filter all received BGP routes.  
Only the routes that match the policy are added to a routing table.
  - e. Run **commit**  
The configuration is committed.
- Configure BGP4+ to filter the routes received from a specific peer (group).
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run the following commands as needed to configure BGP4+ to use different filters to filter routes received from peers:
    - Based on a basic ACL:
      - 1) Run **peer { ipv4-address | ipv6-address | group-name } filter-policy { acl6-number | acl6-name acl6-name } import**

The routes to be received from the specified peer or peer group are filtered.

2) Run **quit**

Return to the BGP view.

3) Run **quit**

Return to the system view.

4) Run **acl ipv6 { name basic-acl6-name basic | [ number ] basic-acl6-number } [ match-order { config | auto } ]**

The ACL view is displayed.

5) Run **rule [ rule-id ] [ name rule-name ] { deny | permit }**

An ACL rule is configured.

When the **rule** command is used to configure a filtering rule for a named ACL, only the configurations specified by **source** and **time-range** take effect.

When a filter-policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route matching the rule will be accepted or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route matching the rule will not be accepted or advertised by the system.
- If the network segment of a route is not within the range specified in an ACL rule, the route will not be accepted or advertised by the system.
- If an ACL does not contain any rules, none of the routes matched against the filter-policy that uses this ACL will be accepted or advertised by the system.
- Routes can be filtered using a blacklist or whitelist:  
If ACL rules are used for matching in configuration order, the system matches the rules in ascending order of their IDs.

Filtering using a blacklist: Configure a rule with a smaller ID and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger ID in the same ACL and specify the action **permit** in this rule to accept or advertise the other routes.

Filtering using a whitelist: Configure a rule with a smaller ID and specify the action **permit** in this rule to permit the routes to be accepted or advertised. Then, configure another rule with a larger ID in the same ACL and specify the action **deny** in this rule to filter out the unwanted routes.

- To use the AS\_Path filter for route filtering, run the **peer { ipv4-address | ipv6-address | group-name } as-path-filter { number | name } import** command.
- To use an IP prefix list for route filtering, run **peer { ipv4-address | ipv6-address | group-name } ip-prefix ipv6-prefix-name import**

- To use a route-policy for route filtering, run **peer { ipv4-address | ipv6-address | group-name } route-policy route-policy-name import**

A peer group and its members can use different inbound policies when receiving routes. This means that each member in a peer group can select its own policy to filter received routes.

- e. Run **commit**

The configuration is committed.

- Limit the number of the routes received from a peer.

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run **peer { group-name | ipv4-address | ipv6-address } route-limit limit [ percentage ] [ alert-only | idle-forever | idle-timeout times ]**

The number of routes that can be accepted from a peer or peer group is set.

The command provides the limit on the number of received routes based on peers. You can configure specific parameters as required to control the action after the number of the routes received from a peer exceeds the threshold.

- **alert-only:** The peer relationship is kept. No more route is accepted after the number of received routes exceeds the threshold, and an alarm is generated and a log is recorded.
- **idle-forever:** The peer relationship is interrupted. The device does not retry setting up a connection. An alarm is generated and a log is recorded. In this case, if you run the **display bgp peer [ verbose ]** command, you can find that the status of the peer is Idle. To restore the BGP connection, run the **reset bgp** command. This parameter is not recommended.
- **idle-timeout:** The peer relationship is interrupted. The device retries setting up a connection after the timer expires. An alarm is generated and a log is recorded. In this case, if you run the **display bgp peer [ verbose ]** command, you can find that the status of the peer is Idle. To restore the BGP connection before the timer expires, run the **reset bgp** command.
- If none of the preceding parameters is set, the peer relationship is disconnected. The device retries setting up a connection after 30 seconds. An alarm is generated and a log is recorded.

 NOTE

If the number of routes received by the local router exceeds the upper limit and the **peer route-limit** command is used for the first time, the local router and its peer reestablish the peer relationship, regardless of whether **alert-only** is set.

- e. Run **commit**

The configuration is committed.

----End

## Configuring BGP4+ Soft Reset

BGP4+ soft reset allows the system to refresh a BGP4+ routing table dynamically without tearing down any BGP4+ connection if routing policies are changed.

### Context

After a BGP4+ routing policy is changed, you can reset the BGP4+ connection for the new policy to take effect immediately. This, however, interrupts the BGP4+ connection temporarily. BGP4+ route-refresh allows the system to refresh the BGP4+ routing table dynamically without tearing down any BGP4+ connection when a policy is changed.

- If a BGP4+ peer supports route-refresh, you can refresh the routing table by running the **refresh bgp** command to perform a soft reset on the BGP4+ connection.
- If a BGP4+ peer does not support route-refresh, you can run the **peer keep-all-routes** command to reserve all the original routes of the peer. In this manner, the routing table can be refreshed without resetting the BGP4+ connection.

### Procedure

- Manually perform a soft reset on a BGP4+ connection with a BGP4+ peer that supports route-refresh.

Perform the following steps on a BGP4+ device:

- a. (Optional) Enable route-refresh.
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **bgp as-number** command to enter the BGP view.
  - iii. Run the **peer { ipv4-address | ipv6-address | group-name } capability-advertise route-refresh** command to enable route-refresh.
  - iv. Run the **commit** command to commit the configuration.
- b. Manually perform a soft reset on a BGP4+ connection.

Run the **refresh bgp ipv6 { all | ipv4-address | ipv6-address | group group-name | external | internal } { export | import }** command to perform a soft reset on a BGP4+ connection.

To perform a soft reset on a BGP4+ connection, run the preceding command in the user view.

The **external** and **internal** parameters indicate soft resets of EBGP and IBGP connections, respectively.

- Retain all routing updates of a BGP4+ peer that does not support route-refresh.

Perform the following steps on a BGP4+ device:

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp *as-number***

The BGP view is displayed.

- c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run **peer { *ipv4-address* | *ipv6-address* | *group-name* } keep-all-routes**

The device is configured to store all routing updates received from its peers.

After this command is used, all routing updates sent by a specified peer are stored, regardless of whether a filtering policy is used. When the local routing policy is changed, the routing updates can be used to regenerate BGP4+ routes.

- e. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring BGP4+ routing policies, check routes that are advertised and received by BGP4+.

## Prerequisites

The routing policies have been configured.

## Procedure

- Run the **display bgp ipv6 network** command to check the routes imported by BGP4+ using the **network** command.
- Run the **display bgp ipv6 routing-table as-path-filter { *as-path-filter-number* | *as-path-filter-name* }** command to check the routes that match a specified AS\_Path filter.
- Run the **display bgp ipv6 routing-table community-filter { { *community-filter-name* | *basic-community-filter-number* } [ **whole-match** ] | *advanced-community-filter-number* }** command to check the routes that match a specified BGP4+ community filter.
- Run the **display ip extcommunity-filter [ *basic-extcomm-filter-num* | *advanced-extcomm-filter-num* | *extcomm-filter-name* ]** command to check the VPN-Target extended community filter information.

- Run the **display ip extcommunity-list soo [ extcomm-filter-name ]** command to check SoO extended community filter information.
- Run the **display bgp ipv6 routing-table peer remoteIpv6Addr { advertised-routes | received-routes } [ statistics ]** command to view the routes advertised to or received from a specified BGP4+ peer.

----End

### 1.1.10.1.7 Configuring BGP4+ XPL

BGP4+ extended routing-policy language (XPL) route-filters help flexibly control route advertisement and acceptance.

#### Usage Scenario

Route-filters can use condition clauses to filter routes and use action clauses to set and modify route attributes. The routes that match the route-filters are accepted or advertised. Route-filters can flexibly control route advertisement and acceptance.

#### Pre-configuration Tasks

Before configuring BGP4+ route-filters, complete the following tasks:

- Configure IP addresses for interfaces and ensure that neighboring devices are reachable at the network layer.
- [Configure basic BGP4+ functions.](#)

#### Using XPL to Filter the BGP4+ Routes to Be Advertised

This section describes how to use XPL to filter the BGP4+ routes to be advertised to BGP4+ peers.

#### Context

A BGP4+ device can use a route-filter to filter the routes to be advertised to all peers or peer groups or only to a specified peer or peer group.

#### Procedure

- Configure a BGP4+ device to filter the routes to be advertised to all peers or peer groups.

Perform the following steps on the BGP4+ device:

- a. Run **system-view**

The system view is displayed.

- b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

- d. Run **route-filter route-filter-name export** [ **direct** | **isis process-id** | **ospfv3 process-id** | **ripng process-id** | **static** | **unr** ]  
The BGP4+ device has been configured to filter the routes to be advertised to all peers or peer groups.
  - e. Run **commit**  
The configuration is committed.
- Configure a BGP4+ device to filter the routes to be advertised to a specified peer or peer group.  
Perform the following steps on the BGP4+ device:
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **bgp as-number**  
The BGP view is displayed.
    - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
    - d. Run **peer ipv6-address route-filter route-filter-name export**  
The BGP4+ device has been configured to filter the routes to be advertised to the specified peer or peer group.
    - e. Run **commit**  
The configuration is committed.

----End

## Using XPL to Filter the BGP4+ Routes to Be Received

BGP4+ uses XPL route-filters to filter the routes to be received from BGP4+ peers.

### Context

A BGP4+ device can use a route-filter to filter the routes to be received from all its peers or peer groups or only from a specified peer or peer group.

### Procedure

- Configure a BGP4+ device to filter the routes to be received from all its peers or peer groups.  
Perform the following steps on the BGP4+ device:
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.

- d. Run **route-filter** *route-filter-name import*  
The device is configured to filter received routes.
  - e. Run **commit**  
The configuration is committed.
- Configure a BGP4+ device to filter the routes to be received from a specified peer or peer group.  
Perform the following steps on the BGP4+ device:
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **bgp** *as-number*  
The BGP view is displayed.
    - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
    - d. Run **peer** *ipv6-address route-filter* *route-filter-name import*  
The device is configured to filter the routes to be received from the specified peer.
    - e. Run **commit**  
The configuration is committed.

----End

### 1.1.10.1.8 Configuring BGP4+ Route Recursion to the Default Route

In case the next hops of BGP4+ routes are not directly reachable, you can configure BGP4+ route recursion to the default route.

#### Usage Scenario

The next hops of BGP4+ routes may not be directly reachable. In this case, recursion is required so that the BGP4+ routes can be used for traffic forwarding. You can configure whether to allow BGP4+ routes to recurse to the default route.

#### Pre-configuration Tasks

Before configuring BGP4+ route recursion to the default route, [configure basic BGP4+ functions](#).

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bgp** *as-number*

The BGP view is displayed.

**Step 3 Run `ipv6-family unicast`**

The IPv6 unicast address family view is displayed.

**Step 4 Run `nexthop recursive-lookup default-route`**

BGP4+ route recursion to the default route is configured.

**Step 5 Run `commit`**

The configuration is committed.

----End

## Verifying the Configuration

After configuring BGP4+ route recursion to the default route, run the **display current-configuration** command in the system view to check whether the function is configured.

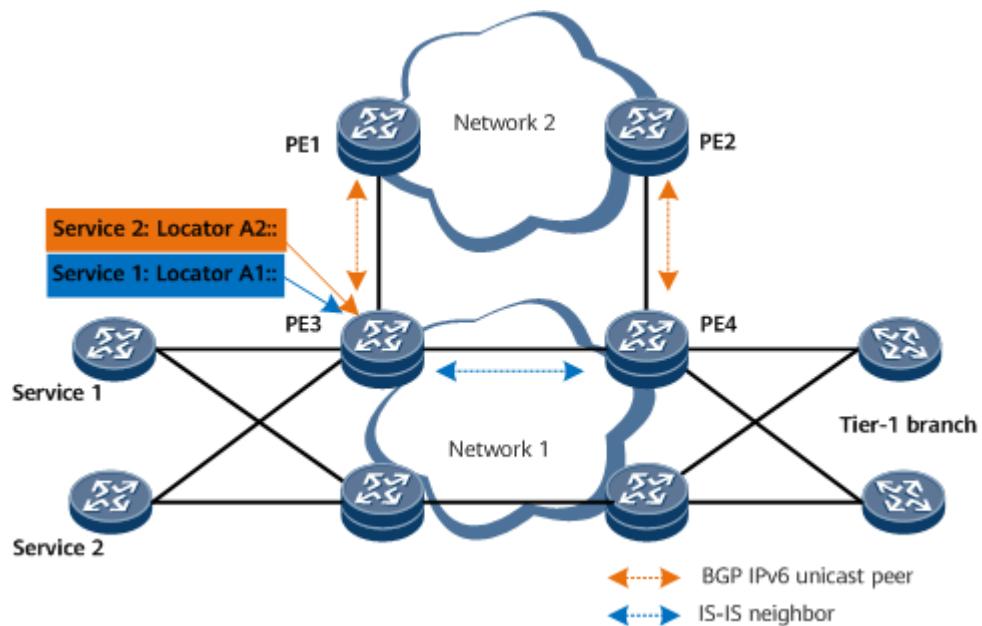
### 1.1.10.1.9 Configuring BGP4+ to Generate Locator Routes

Configuring BGP4+ to generate locator routes allows different services distinguished through locators to be forwarded over different networks.

#### Usage Scenario

As shown in [Figure 1-453](#), user services are transmitted over network 1 and network 2. Network 1 is a user-built network, and network 2 is a carrier network. An IS-IS neighbor relationship is established between PE3 and PE4. A BGP IPv6 unicast peer relationship is established between PE1 and PE3, and between PE2 and PE4. Service 1 and service 2 on the user network are distinguished through locators. To ensure that service 1 and service 2 are forwarded over different networks, you can configure this function on PE3. Suppose that locator A1 is allocated to service 1 and that locator A2 is allocated to service 2. Locator A1 is advertised to PE4 through the IS-IS neighbor relationship, whereas locator A2 is advertised to PE1 through a BGP IPv6 peer relationship and then to the tier-1 branch over network 2. In this case, the service traffic matching locator A1 is forwarded over network 1, whereas the service traffic matching locator A2 is forwarded over network 2.

Figure 1-453 Typical networking



## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run bgp *as-number*

The BGP view is displayed.

### Step 3 Run ipv6-family unicast

The BGP-IPv6 unicast address family view is displayed.

### Step 4 Run segment-routing ipv6 generate-route locator *locator-name* [ route-policy *policy-name* | route-filter *route-filter-name* ]

BGP is configured to generate locator routes.

### Step 5 Run commit

The configuration is committed.

----End

## Verifying the Configuration

After configuring the function, verify the configuration.

Run the **display bgp ipv6 routing-table *ipv6-address*** command to check routes in the BGP4+ routing table.

### 1.1.10.1.10 Configuring BGP4+ Load Balancing

Configuring BGP4+ load balancing better utilizes network resources.

## Usage Scenario

On large networks, there may be multiple valid routes to the same destination. BGP4+, however, advertises only the optimal route to its peers. This may result in traffic imbalance.

Either of the following methods can be used to resolve the traffic imbalance:

- Use BGP4+ route-policies for load balancing. For example, you can use a route-policy to modify the Local\_Pref, AS\_Path, Origin, or MED attribute of BGP4+ routes to control traffic forwarding paths, helping implement load balancing.
- Use multiple paths to implement traffic load balancing. This method requires that multiple equal-cost routes exist and the number of routes allowed to participate in load balancing be set. Load balancing can be implemented globally or for a specified peer or peer group.

## Pre-configuration Tasks

Before configuring BGP4+ load balancing, [configure basic BGP4+ functions](#).

## Procedure

- Configure BGP4+ peer or peer group-based load balancing.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run **peer { ipv4-address | ipv6-address | group-name } load-balancing [ as-path-ignore | as-path-relax ]**  
BGP4+ peer or peer group-based load balancing is configured.

After the **peer load-balancing** command is run, BGP peer-based load balancing is implemented only when the following conditions are met:

- The routes are received from the specified peer or peer group.
- The optimal route and optimal equal-cost routes exist.
- The AS\_Path attribute is the same as that of the optimal route, or **as-path-ignore** or **as-path-relax** is specified in the **peer load-balancing** command.
  - If **as-path-ignore** is specified, the device ignores comparing AS\_Path attributes when selecting routes for load balancing. In this case, routes can participate in load balancing even if their AS\_Path attributes are different.
  - If **as-path-relax** is specified, the device ignores comparing the AS\_Path attributes of the same length when selecting routes for load balancing. In this case, routes cannot participate in load balancing if their AS\_Path attributes are of different lengths. For

example, the AS\_Path attribute of route A is **10**, and the AS\_Path attribute of route B is **10, 20**. Because the two AS\_Path attributes are of different lengths, the two routes cannot participate in load balancing.

e. (Optional) Change load balancing rules.

- To prevent the device from comparing AS\_Path attributes when selecting routes for load balancing, run the **load-balancing as-path-ignore** command.
- To configure the device to ignore comparing the AS\_Path attributes of the same length when selecting routes for load balancing, run the **load-balancing as-path-relax** command.
- To prevent the device from comparing IGP costs when selecting routes for load balancing, run the **load-balancing igr-metric-ignore** command.

 NOTE

The address family views supported by the preceding commands are different. When running any of the commands, ensure that the command is run in the correct address family view.

Change load balancing rules based on network requirements and exercise caution when running the commands.

f. Run **commit**

The configuration is committed.

- Configure global BGP4+ load balancing.
  - Set the maximum number of BGP4+ routes for load balancing.
    - i. Run **system-view**  
The system view is displayed.
    - ii. Run **bgp as-number**  
The BGP view is displayed.
    - iii. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
    - iv. Run **maximum load-balancing [ ebgp | ibgp ] number [ ecmp-nexthop-changed ]**  
The maximum number of BGP4+ equal-cost routes for load balancing is set.
      - **ebgp** indicates that load balancing is implemented only among EBGP routes.
      - **ibgp** indicates that load balancing is implemented only among IBGP routes.
      - If neither **ebgp** nor **ibgp** is specified, both EBGP and IBGP routes can balance traffic, and the number of EBGP routes for load balancing is the same as the number of IBGP routes for load balancing.

 NOTE

If multiple routes with the same destination address exist on the public network, the system selects the optimal route first. If IBGP routes are optimal, only IBGP routes carry out load balancing. If EBGP routes are optimal, only EBGP routes carry out load balancing. This means that load balancing cannot be implemented using both IBGP and EBGP routes with the same destination address.

- v. (Optional) Change load balancing rules.
  - o Run the **load-balancing as-path-ignore** command to prevent the device from comparing AS\_Path attributes when selecting routes for load balancing.
  - o Run the **load-balancing as-path-relax** command to configure the device to ignore comparing the AS\_Path attributes of the same length when selecting routes for load balancing.
  - o Run the **load-balancing igr-metric-ignore** command to prevent the device from comparing IGP costs when selecting routes for load balancing.

 NOTE

The address family views supported by the preceding commands are different. When running any of the commands, ensure that the command is run in the correct address family view.

Change load balancing rules based on network requirements and exercise caution when running the commands.

- vi. Run **commit**

The configuration is committed.

- Set the maximum number of EBGP and IBGP routes for load balancing. This configuration is used in a VPN where a CE is dual-homed to two PEs. When the CE resides in the same AS as only one of the PEs, you can set the maximum number of EBGP and IBGP routes for load balancing so that VPN traffic can be balanced among EBGP and IBGP routes.

- i. Run **system-view**

The system view is displayed.

- ii. Run **bgp as-number**

The BGP view is displayed.

- iii. Run **ipv6-family vpn-instance vpn-instance-name**

The BGP-VPN instance IPv6 address family view is displayed.

- iv. Run **maximum load-balancing eibgp number [ ecmp-nexthop-changed ]**

The maximum number of EBGP and IBGP routes for load balancing is set.

After the **maximum load-balancing eibgp number** command is run on a device, the device, by default, changes the next hop of each BGP4+ route to itself before advertising the route to a peer, regardless of whether the route is to be used for load balancing. However, in RR or BGP confederation scenarios, the device does not change the next hop addresses of non-local routes to be advertised

to a local address. As a result, besides the routes for load-balancing, those routes that are not supposed to participate in load balancing divert traffic to the device, which overburdens the device. To address this problem, you can set **ecmp-nexthop-changed** so that the device changes the next hop of only the BGP4+ routes that are to be used for load balancing to itself before advertising them to peers.

- v. (Optional) Change load balancing rules.
  - o Run the **load-balancing as-path-ignore** command to prevent the device from comparing AS\_Path attributes when selecting routes for load balancing.
  - o Run the **load-balancing as-path-relax** command to configure the device to ignore comparing the AS\_Path attributes of the same length when selecting routes for load balancing.
  - o Run the **load-balancing igr-metric-ignore** command to prevent the device from comparing IGP costs when selecting routes for load balancing.

 NOTE

The address family views supported by the preceding commands are different. When running any of the commands, ensure that the command is run in the correct address family view.

Change load balancing rules based on network requirements and exercise caution when running the commands.

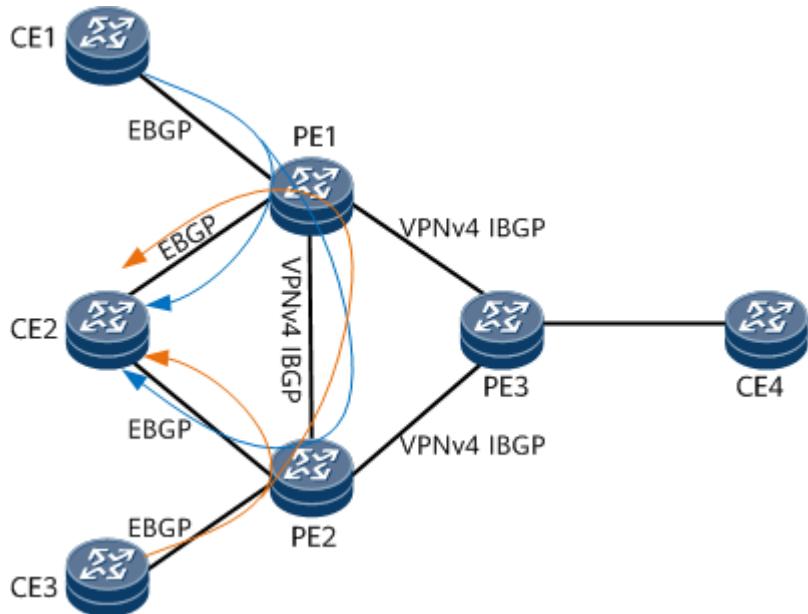
- vi. Run **commit**

The configuration is committed.

- Configure load balancing among VPN unicast routes and leaked routes.

This configuration is mainly used in an EIBGP load balancing scenario where a CE that is single-homed to a PE accesses a CE that is dual-homed to PEs. As shown in [Figure 1-454](#), CE2 is dual-homed to PE1 and PE2. CE1 and CE2 are connected to PE1, and CE2 and CE3 are connected to PE2. When CE1 or CE3 needs to access CE2, you can configure load balancing among VPN unicast routes and leaked routes on PE1 and PE2. In this manner, when CE1 or CE3 accesses CE2, load balancing can be implemented through PE1 and PE2, that is, load balancing among VPN routes and leaked routes.

**Figure 1-454** Load balancing among VPN unicast routes and leaked routes



- i. Run **system-view**  
The system view is displayed.
- ii. Run **ip vpn-instance *vpn-instance-name***  
A VPN instance is created, and its view is displayed.
- iii. Run **route-distinguisher *route-distinguisher***  
An RD is configured for the VPN instance.
- iv. Run **apply-label { per-nexthop | per-route } pop-go**  
A label distribution mode is configured for the current VPN.
- v. Run **quit**  
The VPN instance view is displayed.
- vi. Run **quit**  
The system view is displayed.
- vii. Run **bgp *as-number***  
The BGP view is displayed.
- viii. Run **ipv6-family vpn-instance *vpn-instance-name***  
The BGP-VPN instance IPv6 address family view is displayed.
- ix. Run **load-balancing local-learning cross**  
Load balancing among VPN unicast routes and leaked routes is configured.  
The **load-balancing local-learning cross** command can be run only after the **apply-label { per-nexthop | per-route } pop-go** command is run in the corresponding address family view of the VPN instance. If the **apply-label { per-nexthop | per-route } pop-go** command is not run, routing loops may occur between PEs.  
The **load-balancing local-learning cross** command is mutually exclusive with the **segment-routing ipv6 locator evpn, segment-**

**routing ipv6 locator, vxlan vni, and evpn mpls routing-enable** commands.

x. Run **commit**

The configuration is committed.

----End

## Follow-up Procedure

When BGP4+ routes carrying the link bandwidth extended community attribute are available for load balancing and they all recurse to IP routes or tunnels, you can run the **load-balancing ucmp** command in the BGP-IPv6 unicast address family view to implement unequal-cost load balancing among BGP4+ routes based on the link bandwidth extended community attribute. With this function, when there are multiple egress devices to the destination, unequal-cost load balancing can be implemented based on the actual bandwidth capability of each egress device. The methods of configuring the link bandwidth extended community attribute are as follows:

- Use **XPL to configure the link bandwidth extended community attribute**.
- Run the **peer generate-link-bandwidth** command to configure the local device to obtain the link bandwidth of a specified directly connected EBGP peer and generate the extended community attribute accordingly.

## Verifying the Configuration

After the configuration is complete, check whether the configuration has taken effect.

Run the **display bgp ipv6 routing-table ipv6-address prefix-length** command to check routes in the BGP4+ routing table.

### 1.1.10.1.11 Configuring BGP4+ to Generate a Summary Default Route

You can configure BGP4+ to generate a summary default route and then determine whether to advertise the default route to a peer by using a route-policy.

## Usage Scenario

On the network shown in [Figure 1-455](#), a VPNv6 peer relationship is established between PE1 and PE2. In addition, an EBGP-VPN peer relationship is established between PE2 and the core network device, and another EBGP-VPN peer relationship is established between PE1 and CE1. The core network device advertises routes to PE2 through the EBGP-VPN peer relationship. PE2 then advertises the received routes to PE1 through the VPNv6 peer relationship. Upon receipt, PE1 leaks the routes to its VPN instance. A default route can be generated only if routes with a specific prefix exist among the leaked routes. You can perform this configuration task so that PE1 advertises a default route to CE1 based on an IP prefix list filter. In this way, traffic corresponding to the routes that do not match the filter is not diverted to PE1, thereby conserving PE1's bandwidth resources.

Figure 1-455 Typical networking



## Pre-configuration Tasks

Before configuring BGP4+ to generate a summary default route, [configure basic BGP4+ functions](#).

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run ip ipv6-prefix *ipv6-prefix-name* [ **index** *index-number* ] **matchMode** *ipv6-address masklen* [ **match-network** ] [ **greater-equal** *greater-equal-value* ] [ **less-equal** *less-equal-value* ]

An IPv6 prefix list is configured.

### Step 3 Run bgp *as-number*

The BGP view is displayed.

### Step 4 Run ipv6-family unicast

The IPv6 unicast address family view is displayed.

### Step 5 Run aggregate default-route origin-ipv6-prefix *ipv6-prefix-name* [ **attribute-policy** *attribute-policy-name* ]

BGP is enabled to generate a summary default route based on an IPv6 prefix list.

### Step 6 Run commit

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify the configuration.

Run the **display bgp ipv6 routing-table** command to check information about BGP4+ summary default routes.

### 1.1.10.1.12 Configuring BGP4+ Connection Parameters

By configuring BGP4+ connection parameters, you can optimize BGP4+ network performance.

## Usage Scenario

BGP4+ can use various timers to minimize the impact of interface flapping or route flapping.

- Timers for BGP4+ Peers

After establishing a BGP4+ connection, two peers periodically send Keepalive messages to each other to detect the status of the BGP4+ peer relationship. If the router does not receive any Keepalive message or any other type of packets from its peer within a time of period called holdtime period, it considers the BGP4+ connection closed.

When establishing a BGP4+ connection with a peer, the router compares the hold time values of the two peers. The smaller hold time is used as the negotiated hold time. If the negotiation result is 0, no Keepalive message is sent and whether the Hold timer expires is not detected.

### NOTICE

If the value of the timer changes, the BGP4+ connection is interrupted for a short time because the peers need to renegotiate with each other.

- BGP4+ ConnectRetry Timer

After BGP4+ initiates a TCP connection, the ConnectRetry timer will be stopped if the TCP connection is established successfully. If the first attempt to establish a TCP connection fails, BGP4+ re-establishes the TCP connection after the ConnectRetry timer expires. Setting a short ConnectRetry interval reduces the period BGP4+ waits between attempts to establish a TCP connection, which speeds up the establishment of the TCP connection. Setting a long connectRetry interval suppresses routing flapping caused by peer relationship flapping.

## Pre-configuration Tasks

Before configuring BGP4+ connection parameters, [configure basic BGP4+ functions](#).

## Configuring Timers for BGP4+ Peers

Configuring timers properly improves network performance. Changing the timeout periods of timers, however, will interrupt peer relationships.

## Context

### NOTICE

Changing the timeout period of a timer using the **peer timer** command will interrupt peer relationships between routers. Therefore, exercise caution when changing the timeout period of a timer.

Perform the following steps on a BGP4+ device:

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run the **peer { ipv6-address | group-name } timer keepalive keepalive-time hold hold-time [ min-holdtime min-hold-value ]** command to configure the interval at which Keepalive messages are sent and the hold time for a peer or peer group. Alternatively, run the **peer ipv6-address timer send-hold send-hold-time** command to configure the hold time during which a peer does not proactively disconnect from the peer end.

The value of *hold-time* must be at least three times the value of *keepalive-time*.

#### NOTE

You are advised to configure *hold-time* according to the total number of peers in all BGP address families. The more peers, the larger the recommended minimum hold time value. You can adjust the hold time based on [Table 1-125](#).

**Table 1-159** Mapping between the total number of BGP peers in all address families and the recommended minimum hold time

| Total Number of Peers        | Recommended Minimum Hold Time |
|------------------------------|-------------------------------|
| 0–100                        | 20s                           |
| 101–200                      | 30s                           |
| 201–300                      | 45s                           |
| 301–400                      | 60s                           |
| 401–500                      | 75s                           |
| Greater than or equal to 501 | 90s                           |

Avoid the following situations when setting values for the three timers:

- The values of *keepalive-time* and *hold-time* are both set to 0. In this case, the BGP timers become invalid, and BGP cannot detect link faults based on the timers.
- The *hold-time* value is much greater than the *keepalive-time* value, for example, **timer keepalive 1 hold 65535** is configured. In this case, link faults cannot be detected in time.
- The *keepalive-time* value is set to 0. In this case, the keepalive timer does not start. As a result, the *send-hold-time* function does not take effect.

#### Step 4 Run **commit**

The configuration is committed.

----End

### Enabling Fast EBGP Connection Reset

After fast EBGP connection reset is enabled, BGP4+ can rapidly detect EBGP link faults and reset BGP4+ connections on interfaces.

## Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bgp as-number**

The BGP view is displayed.

#### Step 3 Run **ebgp-interface-sensitive**

Fast EBGP connection reset is enabled.

If the interface on which a BGP4+ connection is established alternates between up and down, you can run the **undo ebgp-interface-sensitive** command to prevent repeated reestablishment and deletion of the BGP4+ session caused by route flapping. This saves network bandwidth.

#### Step 4 Run **commit**

The configuration is committed.

----End

### Configuring a BGP4+ ConnectRetry Timer

You can control the speed at which BGP4+ peer relationships are established by changing the BGP4+ ConnectRetry timer value.

## Context

After BGP4+ initiates a TCP connection, the ConnectRetry timer will be stopped if the TCP connection is established successfully. If the first attempt to establish a TCP connection fails, BGP4+ re-establishes the TCP connection after the ConnectRetry timer expires.

- Setting a short ConnectRetry interval reduces the period BGP4+ waits between attempts to establish a TCP connection, which speeds up the establishment of the TCP connection.
- Setting a long connectRetry interval suppresses routing flapping caused by peer relationship flapping.

A ConnectRetry timer can be configured either for all peers or peer groups, or for a specific peer or peer group. A ConnectRetry timer configured for a specific peer takes precedence over that configured for the peer group of this peer. In addition, a ConnectRetry timer configured for a specific peer or peer group takes precedence over that configured for all peers or peer groups.

## Procedure

- Configure a BGP4+ ConnectRetry timer for all peers or peer groups.

Perform the following steps on a BGP4+ router:

  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp as-number**

The BGP view is displayed.
  - c. Run **timer connect-retry connect-retry-time**

A BGP4+ ConnectRetry timer is configured for all peers or peer groups.
  - d. Run **commit**

The configuration is committed.
- Configure a ConnectRetry timer for a specific peer or peer group.

Perform the following steps on a BGP4+ router:

  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp as-number**

The BGP view is displayed.
  - c. Run **peer { group-name | ipv6-address } timer connect-retry connect-retry-time**

A ConnectRetry timer is configured for a specified peer or peer group.

The ConnectRetry timer configured for a peer takes precedence over that configured for a peer group. The ConnectRetry timer configured for a peer or peer group takes precedence over that configured for all peers or peer groups.
  - d. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring BGP4+ connection parameters, check information about BGP4+ peers or peer groups.

## Prerequisites

BGP4+ connection parameters have been configured.

## Procedure

- Run the **display bgp ipv6 peer [ *ipv6-address* ] verbose** command to check information about BGP4+ peers.

----End

### 1.1.10.1.13 Configuring BGP4+ RRs

BGP4+ RRs avoid fully meshed connections between multiple IBGP peers, which reduces network costs.

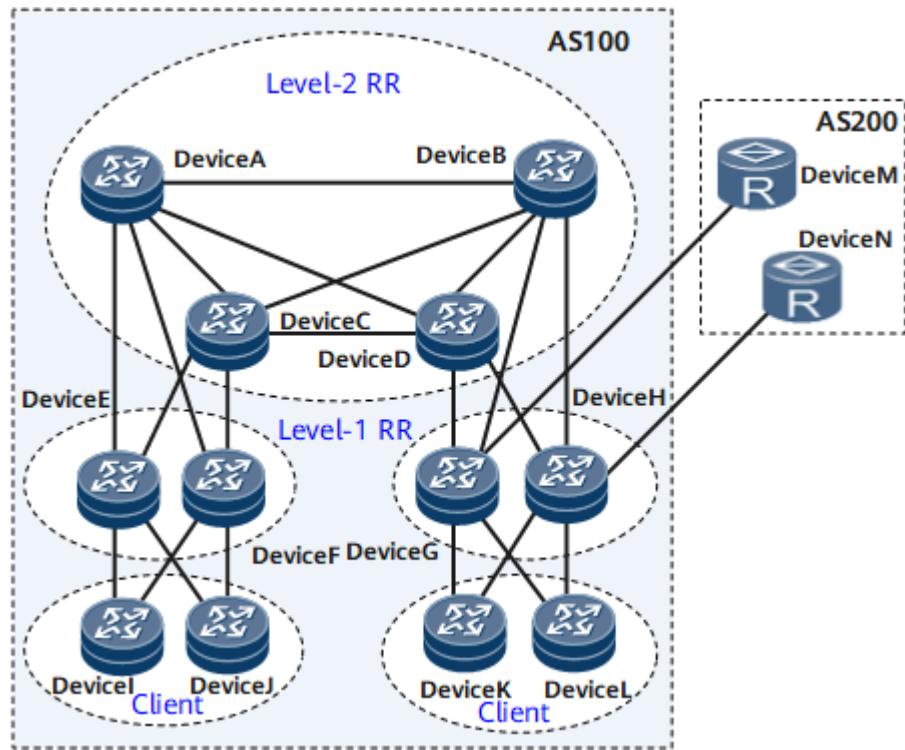
## Usage Scenario

Fully meshed connections need to be established between IBGP peers to ensure the connectivity between IBGP peers. If there are n routers in an AS,  $n \times (n-1)/2$  IBGP connections need to be established. When there are a lot of IBGP peers, network resources and CPU resources are greatly consumed. Route reflection can solve the problem.

Using RRs reduces the total number of IBGP connections. On a large network, however, multiple RRs need to be configured to reduce the number of clients of each RR. Therefore, there are still a large number of IBGP connections on the network because fully meshed connections need to be established between the RRs. To further reduce the number of IBGP connections, multi-level BGP4+ RR networking is introduced.

**Figure 1-456** shows a typical BGP4+ hierarchical RR networking. Device A, Device B, Device C, and Device D function as level-2 RRs; Device E, Device F, Device G, and Device H function as level-1 RRs and the clients of level-2 RRs. Level-2 RRs are not the clients of any RR and therefore must be fully meshed. Level-1 RRs function as the clients of level-2 RRs and do not need to be fully meshed.

Figure 1-456 Networking with hierarchical RRs



## Pre-configuration Tasks

Before configuring BGP4+ RRs, [configure basic BGP4+ functions](#).

## Configuring an RR and Specifying Its Clients

RRs reflect routes between clients, and therefore IBGP connections do not need to be established between the clients.

## Context

In an AS, one router functions as an RR, and the other routers function as its clients. The clients establish IBGP connections with the RR. The RR and its clients form a cluster. The RR transmits or reflects routes among clients, but the clients do not need to establish any IBGP connections between each other.

An RR is easy to configure because it needs to be configured only on the router that functions as a reflector and clients do not need to know that they are clients.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Run **peer { ipv6-address | ipv4-address | group-name } reflect-client**

An RR and its clients are configured.

The router where the **peer reflect-client** command is run functions as the RR, and specified peers or peer groups function as the clients.

 **NOTE**

**reflect-client** configured in an address family is valid in this family address and cannot be inherited by other address families.

**Step 5** Run **commit**

The configuration is committed.

----End

### (Optional) Disabling Route Reflection Between Clients Through the RR

If the clients of an RR are fully meshed, you can disable route reflection between the clients through the RR. This reduces the memory overhead.

## Context

On some networks, if IBGP peer connections have already been established among clients of an RR, they can exchange routing information directly. In this case, route reflection between the clients through the RR is unnecessary and consumes bandwidth resources. In this case, you can disable route reflection between the clients through the RR. This reduces the memory overhead on the network.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Run **undo reflect between-clients**

Route reflection between clients through the RR is disabled.

If the clients of an RR have been fully meshed, you can run the **undo reflect between-clients** command to disable route reflection between the clients through the RR. The **undo reflect between-clients** command is run only on RRs.

**Step 5** Run **commit**

The configuration is committed.

----End

## (Optional) Configuring a Cluster ID for RRs

If a cluster has multiple RRs, you can configure the same cluster ID for these RRs to prevent routing loops.

### Context

Under some circumstances, more than one RR needs to be configured in a cluster to improve network reliability and prevent single points of failure. The same cluster ID needs to be configured for all the RRs in the cluster to reduce the number of routes received by each RR. This reduces network cost.

#### NOTE

To allow clients to receive routes reflected by RRs, ensure that the cluster ID of the RRs is different from the router ID of any client. If the cluster ID of the RRs is the same as the router ID of a client, the client will discard received routes.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **bgp as-number**

The BGP view is displayed.

#### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

#### Step 4 Run **reflector cluster-id { cluster-id-value | cluster-id-ipv4 }**

A cluster ID is configured for RRs.

If a cluster has multiple RRs, you can use this command to set the same cluster ID for these RRs.

The **reflector cluster-id** command is run only on RRs.

#### Step 5 Run **commit**

The configuration is committed.

----End

## (Optional) Preventing BGP4+ from Adding Routes to the IP Routing Table

Disabling BGP4+ route delivery to the IP routing table on an RR can prevent traffic from being forwarded by the RR, improving route advertisement efficiency.

### Context

In most cases, BGP4+ routes are added to the IP routing table on the router for traffic forwarding. If the router does not need to forward traffic, prevent the router from adding BGP4+ routes to the IP routing table.

Preventing a device from adding BGP4+ routes to the IP routing table is mainly used in RR scenarios. In an AS, an RR transmits routes and forwards traffic. If an

RR is connected to many clients and non-clients, route transmission will consume a lot of CPU resources of the RR and cause the RR to be unable to implement traffic forwarding. To improve route transmission efficiency, prevent the RR from adding BGP4+ routes to the IP routing table.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

### Step 4 Run **routing-table rib-only [ route-policy route-policy-name | route-filter route-filter-name ]**

BGP4+ is prevented from adding routes to the IP routing table.

If **route-policy route-policy-name** or **route-filter route-filter-name** is configured in the **routing-table rib-only** command, routes matching the policy are not added to the IP routing table, and routes that do not match the policy are added to the IP routing table, with the route attributes unchanged.

### Step 5 Run **commit**

The configuration is committed.

----End

## (Optional) Enabling an RR to Modify Route Attributes Using an Export Policy

Enabling an RR to modify route attributes using an export policy can change BGP4+ route selection results.

## Context

The route attributes on the RR cannot be modified using the export policy because it may cause routing loops. By default, the RR is disabled from modifying the route attributes based on the export policy. However, if you need to re-plan the network traffic, you can enable the RR to modify route attributes based on an export policy.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

### Step 4 Run **reflect change-path-attribute**

The RR is configured to modify the path attributes of BGP4+ routes based on an export policy.

After the **reflect change-path-attribute** command is run on an RR, the configuration of modifying path attributes based on an export policy takes effect immediately. After the command is run, the following configurations can take effect:

- **apply as-path**: This command configures the device to modify the AS\_Path attribute of BGP4+ routes.
- **apply comm-filter delete**: This command configures the device to delete community attributes of BGP4+ routes.
- **apply community**: This command configures the device to modify community attributes of BGP4+ routes.
- **apply large-community**: This command configures the device to modify the Large-Community attribute of BGP4+ routes.
- **apply cost**: This command configures the device to modify the cost (MED) values of BGP4+ routes.
- **apply ipv6 next-hop**: This command configures the device to modify the next-hop IP addresses of BGP4+ routes.
- **apply local-preference**: This command configures the device to modify the local preferences of BGP4+ routes.
- **apply origin**: This command configures the device to modify the origin attribute of BGP4+ routes.
- **apply extcommunity**: This command configures the device to modify the VPN target extended community attribute of BGP4+ routes.

#### NOTE

After the **reflect change-path-attribute** command is run on an RR, the **peer route-policy export** command takes precedence over the **peer next-hop-invariable** and **peer next-hop-local** commands.

### Step 5 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring BGP4+ RRs, check information about BGP4+ routes and peer groups.

## Prerequisites

BGP4+ RRs have been configured.

## Procedure

- Run the **display bgp ipv6 peer [ *ipv6-address* ] verbose** command to check information about peers and check whether the relationship with the RR is successfully established.
- Run the **display bgp ipv6 routing-table** command to check whether there are routes reflected by an RR.

----End

### 1.1.10.1.14 Configuring a BGP4+ Confederation

On a large BGP4+ network, configuring a BGP4+ confederation reduces the number of IBGP connections and simplifies routing policy management, which increases the route advertisement efficiency.

## Usage Scenario

A confederation is a solution to the increasing number of IBGP connections in an AS. The confederation divides an AS into multiple sub-ASs. In each sub-AS, IBGP peer relationships are set up or an RR is configured on one of the IBGP peers. EBGP connections are set up between sub-ASs.

## Pre-configuration Tasks

Before configuring a BGP4+ confederation, complete the following tasks:

- Configure link layer protocol parameters and assigning IP addresses to the interfaces to ensure that the status of the link layer protocol of the interface is Up.
- Configure basic BGP4+ functions.**

## Procedure

- Configure a BGP4+ confederation.

Perform the following steps on the BGP4+ router:

- Run **system-view**

The system view is displayed.

- Run **bgp *as-number***

The BGP view is displayed.

- Run **confederation id *as-number***

The confederation ID is set.

- Run **confederation peer-as { *as-number* } &<1-32>**

The number of the sub-AS of other EBGP peers is set.

The parameter *as-number* used is valid only in the confederation.

The **confederation id** and **confederation peer-as** commands must be run for all the EBGP peers in a confederation, and the EBGP peers must have the same confederation ID.

 NOTE

The old speaker supporting 2-byte AS numbers and the new speaker supporting 4-byte AS numbers cannot exist in the same confederation. Otherwise, routing loops may occur because AS4\_Path does not support confederations.

e. Run **commit**

The configuration is committed.

- Configure the compatibility of the confederation.

If the confederation implementation of some routers do not comply with the standard protocol, you can perform the following steps on BGP4+ routers for compatibility:

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **confederation nonstandard**

d. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the following command to check the previous configuration.

- Run the **display bgp ipv6 peer [ ipv6-address ] verbose** command to check detailed information about the peer.
- Run the **display bgp ipv6 routing-table [ ipv6-address prefix-length ]** command to check information about the BGP4+ routing table.

### 1.1.10.1.15 Configuring BGP4+ Community Attributes

The community attribute is used to simplify route-policy management.

## Context

The community attribute is used to simplify the application, maintenance, and management of route-policies. With the community attribute used, a group of BGP4+ peers in multiple ASs can share a route-policy. Before advertising a route carrying a community attribute to peers, a BGP4+ device can be configured to change the original community attribute of this route. Community attributes are route attributes, which are transmitted between BGP4+ peers, and the transmission is not restricted within an AS.

## Pre-configuration Tasks

Before configuring BGP4+ community attributes, complete the following task:

- [1.1.10.1.3 Configuring Basic BGP4+ Functions](#)

## Configuring a Community Attribute-Related Policy

A policy that references a community attribute needs to be configured before the community attribute is set for routing information.

### Procedure

- Configure a community attribute-based route-policy.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **route-policy route-policy-name matchMode node node**

A route-policy is created, and the route-policy view is displayed.
  - c. (Optional) Configure if-match clauses for the route-policy. Community attributes can be added only to the routes that match if-match clauses, and the community attributes of only the routes that match the if-match clauses can be modified. For details about the configuration, see [\(Optional\) Configuring an if-match Clause](#).
  - d. Configure community or extended community attributes for BGP4+ routes. This section describes only common configurations. For details, see [\(Optional\) Configuring an apply Clause](#).
    - To configure a community attribute for BGP4+ routes, run the **apply community { cmntyValue | cmntyNum | internet | no-advertise | no-export | no-export-subconfed } &<1-32> [ additive ]** command.
- **NOTE**

A maximum of 32 community attributes can be configured using this command at a time.

  - To configure the BGP4+ VPN-Target extended community attribute, run the **apply extcommunity { rt extCmntyValue } &<1-16> [ additive ]** command.
  - To configure the SoO extended community attribute for BGP4+ routes, run the **apply extcommunity soo { site-of-origin } &<1-16> additive** command.
- Run **commit**

The configuration is committed.
- Configure a community attribute-based route-filter.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **xpl route-filter route-filter-name** or **edit xpl route-filter route-filter-name**

A route-filter is created, and the route-filter view is displayed.
  - c. (Optional) Configure XPL common and condition clauses. The community attributes of routes can be added or modified only if the routes match specified clauses. For details about the configuration, see [Common Clauses](#) and [Condition Clauses](#) in "XPL Configuration."

- d. Configure community or extended community attributes for BGP4+ routes.
  - Configure community attributes for BGP4+ routes. For configuration details, see [Action Clauses Used to Set Community Attributes for BGP Routes](#) in "XPL Configuration."
  - Configure the Link Bandwidth extended community attribute for BGP4+ routes. For configuration details, see [Setting a Link Bandwidth Extended Community Attribute for BGP Routes](#) in "XPL Configuration."
- e. Run **commit**

The configuration is committed.

----End

## Configuring Community Attribute Advertisement

A community attribute defined in a rout-policy takes effect only after community attribute advertisement is configured.

### Procedure

- Use a route-policy to define specific community attributes before configuring BGP4+ community attribute advertisement.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp as-number**

The BGP view is displayed.
  - c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.
  - d. Run **peer ipv6-address route-policy route-policy-name export**

An export route-policy is configured.
  - e. Run **peer ipv6-address advertise-community**

The device is configured to advertise standard community attributes to the specified peer.
  - f. Run **commit**

The configuration is committed.
- Use a route-filter to define specific community attributes when configuring BGP4+ community attribute advertisement.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp as-number**

The BGP view is displayed.

- c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
- d. Run **peer *ipv6-address route-filter route-filter-name export***  
An export route-filter is configured.
- e. Run **peer *ipv6-address advertise-community***  
The device is configured to advertise standard community attributes to the specified peer.
- f. Run **commit**  
The configuration is committed.

----End

## Configuring the Device to Advertise Extended Community Attributes

The extended community attributes defined in a route-policy take effect only after the extended community attributes are advertised.

### Procedure

- Use a route-policy to define specific extended community attributes before configuring BGP4+ to advertise the extended community attributes.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp *as-number***  
The BGP view is displayed.
  - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
  - d. Run **peer { *ipv6-address | ipv6-address | group-name* } route-policy *route-policy-name export***  
An export route-policy is configured.
  - e. Run **peer { *ipv4-address | ipv6-address | group-name* } advertise-ext-community**  
BGP4+ is configured to send routes with extended community attributes to the specified peer.

If a peer only needs to accept routes but not extended community attributes, you can run the **peer discard-ext-community** command on the peer to discard the extended community attributes carried in received routes. If the peer only needs to discard the RPKI BGP origin AS validation result, specify **origin-as-validation** in the command.

- f. (Optional) Run **peer { *ipv4-address | ipv6-address | group-name* } advertise ebgp link-bandwidth**

The device is configured to advertise the link bandwidth extended community attribute to the specified EBGP peer.

g. (Optional) Run **peer { ipv4-address | ipv6-address } advertise link-bandwidth transitive**

The device is configured to convert the link bandwidth extended community attribute (optional non-transitive) carried in BGP4+ routes into an optional transitive attribute before advertising the routes to the specified peer. If the device changes the next-hop address of a received route carrying the link bandwidth extended community attribute to its own address, the device deletes this attribute before advertising the route to the specified peer.

h. Run **commit**

The configuration is committed.

- Use a route-filter to define specific extended community attributes before configuring BGP4+ to advertise the extended community attributes.

a. Run **system-view**

The system view is displayed.

b. Run **bgp as-number**

The BGP view is displayed.

c. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

d. Run **peer { ipv4-address | ipv6-address | group-name } route-filter route-filter-name export**

An export route-filter is configured.

e. Run **peer { ipv4-address | ipv6-address | group-name } advertise-ext-community**

BGP4+ is configured to send routes with extended community attributes to the specified peer.

If a peer only needs to accept routes but not extended community attributes, you can run the **peer discard-ext-community** command on the peer to discard the extended community attributes carried in received routes. If the peer only needs to discard the RPKI BGP origin AS validation result, specify **origin-as-validation** in the command.

f. (Optional) Run **peer { ipv4-address | ipv6-address | group-name } advertise ebgp link-bandwidth**

The device is configured to advertise the link bandwidth extended community attribute to the specified EBGP peer.

g. (Optional) Run **peer { ipv4-address | ipv6-address } advertise link-bandwidth transitive**

The device is configured to convert the link bandwidth extended community attribute (optional non-transitive) carried in BGP4+ routes into an optional transitive attribute before advertising the routes to the specified peer. If the device changes the next-hop address of a received route carrying the link bandwidth extended community attribute to its own address, the device deletes this attribute before advertising the route to the specified peer.

h. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring BGP4+ community attributes, verify the configuration.

## Prerequisites

A BGP4+ community attribute has been configured.

## Procedure

- Run the **display bgp routing-table network [ mask | mask-length ]** command to check information about BGP4+ routes.

----End

### 1.1.10.1.16 Configuring Prefix-based BGP4+ ORF

After prefix-based BGP4+ outbound route filtering (ORF) is configured, the local device sends its prefix-based import policy to a peer so that the peer filters routes during route advertisement.

## Usage Scenario

If a device expects to receive only required routes from a remote device, and the remote device cannot maintain a separate export policy for each connected peer, you can configure prefix-based ORF to implement on-demand route advertisement.

## Pre-configuration Tasks

Before configuring prefix-based BGP4+ ORF, complete the following tasks:

- [Configure basic BGP4+ functions.](#)
- [Configure an IPv6 prefix list.](#)

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** (Optional) Run **orf-limit limit-value**

The maximum number of ORFs that can be accepted from a peer is set.

**Step 4** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 5** Run **peer { group-name | ipv4-address | ipv6-address } ipv6-prefix ipv6-prefix-name import**

An import route-policy that is based on an IPv6 prefix list is configured for a peer or peer group.

**Step 6** Run **peer { group-name | ipv4-address | ipv6-address } capability-advertise orf ipv6-prefix { both | receive | send }**

Prefix-based ORF is configured for a BGP4+ peer or peer group.

**Step 7** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp ipv6 peer [ ipv4-address | ipv6-address ] verbose** command to check detailed information about BGP4+ peers.
- Run the **display bgp ipv6 peer [ ipv4-address | ipv6-address ] orf ipv6-prefix** command to check prefix-based ORF information received from a specified peer.

### 1.1.10.1.17 Configuring the BGP4+ Add-Path Function

BGP4+ Add-Path allows a device to send multiple routes with the same prefix to a specified IBGP peer. These routes can back up each other or load-balance traffic, which improves network reliability.

## Pre-configuration Tasks

Before configuring the BGP4+ Add-Path function, complete the following task:

- Configure basic BGP4+ functions.**

## Procedure

- Perform the following steps on a device that needs to send Add-Path routes:
  - Run **system-view**  
The system view is displayed.
  - (Optional) Run **route-policy route-policy-name matchMode node node**  
A route-policy is created, and the route-policy view is displayed.
  - (Optional) Run **quit**  
Return to the system view.
  - (Optional) Run **xpl route-filter route-filter-name**  
A route-filter is created, and the route-filter view is displayed.
  - (Optional) Run **end-filter**  
Return to the system view.

- f. Run **bgp as-number**  
The BGP view is displayed.
- g. Run **peer { ipv4-address | ipv6-address | peerGroupName } as-number as-number**  
An IP address and AS number are specified for a peer.
- h. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
- i. Run **peer { ipv4-address | ipv6-address | group-name } enable**  
The IPv6 peer is enabled.
- j. Run **bestroute add-path path-number path-number**  
BGP Add-Path is enabled, and the number of preferred routes is specified.
- k. Run **peer { ipv4-address | ipv6-address | group-name } capability-advertise add-path send**  
The device is enabled to send Add-Path routes to the specified peer.
- l. Run **peer { peerIpv4Addr | peerIpv6Addr | groupName } advertise add-path path-number number { route-policy route-policy-name | route-filter route-filter-name }**  
The maximum number of preferred routes to be advertised to a specified peer is specified.

#### NOTE

To configure **route-policy route-policy-name**, you need to enter the route-policy view.

To configure **route-filter route-filter-name**, you need to enter the route-filter view.

- m. Run **commit**  
The configuration is committed.
- Perform the following steps on a device that needs to accept Add-Path routes:
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **bgp as-number**  
The BGP view is displayed.
    - c. Run **ipv6-family unicast**  
The IPv6 unicast address family view is displayed.
    - d. Run **peer { ipv4-address | ipv6-address | group-name } capability-advertise add-path receive**  
The device is enabled to accept the Add-Path routes received from the specified peer.
    - e. Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After completing the configurations, check the configurations on the device that advertises Add-Path routes.

Run the **display bgp peer verbose** command to check the status of BGP4+ Add-Path.

### 1.1.10.1.18 Configuring BFD for BGP4+

BFD for BGP4+ provides BGP4+ with a fast fault detection mechanism, which speeds up network convergence.

#### Usage Scenario

BFD is dedicated to fast detection of forwarding faults to ensure QoS of voice, video, and other video-on-demand services on a network. It enables service providers to provide users with required VoIP and other real-time services of high availability and scalability.

BGP4+ periodically sends Keepalive messages to its peers to monitor peer relationships. This process takes more than one second. When traffic is transmitted at gigabit rates, fault detections that take so long add up and eventually cause packet loss. Because of this issue, the system cannot meet the high reliability requirements of the carrier-class network.

To address this problem, configure BFD for BGP4+. BFD for BGP4+ detects faults on links between BGP4+ peers within milliseconds. If a fault is detected, it notifies BGP of the fault. Therefore, BGP4+ routes can undergo fast convergence.

#### Pre-configuration Tasks

Before configuring BFD for BGP4+, complete the following tasks:

- Configure link layer protocol parameters and IP addresses for interfaces to ensure that the link layer protocol on the interfaces is Up.
- Configure basic BGP4+ functions.**

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bfd**

BFD is enabled globally.

##### Step 3 Run **quit**

Return to the system view.

**Step 4** Run **bgp as-number**

The BGP view is displayed.

**Step 5** Run **peer { group-name | ipv6-address } bfd enable [ single-hop-prefer | { per-link one-arm-echo } ] [ compatible ]**

BFD is configured for a peer or peer group, and default BFD parameters are used to establish a BFD session.

**Step 6** (Optional) Run **peer { group-name | ipv6-address } bfd { min-tx-interval min-tx-interval | min-rx-interval min-rx-interval | detect-multiplier multiplier }<sup>\*</sup>**

Various parameters used for establishing BFD sessions are set.

If BFD is configured on a peer group, the peers that are in the peer group but are not configured with the **peer bfd block** command establish BFD sessions.

 NOTE

- A BFD session can be established only when the corresponding BGP session is in the Established state.
- The configuration of a peer takes precedence over that of its peer group. If BFD is not configured on a peer while its peer group is enabled with BFD, the peer inherits the BFD configurations of its peer group.

**Step 7** (Optional) Run **peer ipv6-address bfd block**

A peer is prevented from inheriting the BFD configuration of its peer group.

If BFD is enabled on a peer group, all its peers will inherit the BFD configuration of the peer group and create a BFD session. If you do not want a peer to inherit the BFD configuration of its peer group, you can run this command.

 NOTE

The **peer ipv6-address bfd block** command and the **peer ipv6-address bfd enable** command are mutually exclusive. After being configured with the **peer bfd block** command, a device automatically deletes the BFD session established with a specified peer.

**Step 8** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp ipv6 bfd session { [ vpnv6 vpn-instance vpn-instance-name ] peer ipv6-address | all }** command to check BFD sessions established by BGP4+.
- Run the **display bgp [ vpnv6 vpn-instance vpn-instance-name ] peer [ ipv6-address ] [ verbose ]** command to check BGP4+ peers.

### 1.1.10.1.19 Configuring BGP4+ Auto FRR

As a protection measure against link faults, BGP4+ Auto Fast Reroute (FRR) is applicable to the network topology with primary and backup links. BGP4+ Auto FRR is applicable to services that are very sensitive to packet loss and delays.

#### Usage Scenario

As networks evolve continuously, voice, on-line video, and financial services raise increasingly high requirements for real-time performance. Usually, primary and backup links are deployed on a network to ensure the stability of these services. In a traditional forwarding mode, the router selects a route out of several routes that are bound for the same destination network as the optimal route and delivers the route to the FIB table to guide data forwarding. If the optimal route fails, the router has to wait for route convergence to be completed before reselecting an optimal route and delivering it to the FIB table. In this case, services are interrupted for a long time, unable to meet service requirements.

BGP Auto FRR allows a device to select the optimal route from the routes that are bound for the same destination network and automatically add information about the suboptimal route to the backup forwarding entry of the optimal route. If the primary link fails, the device quickly switches traffic to the backup link. The switchover does not depend on route convergence and can be performed within sub-seconds, greatly reducing service interruption time.

#### Pre-configuration Tasks

Before configuring BGP4+ Auto FRR, complete the following tasks:

- Configure static route or an IGP to ensure that IP routes between routers are reachable.
- Configure basic BGP4+ functions.**

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bgp as-number**

The BGP view is displayed.

##### Step 3 Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

##### Step 4 Perform either of the following operations to enable BGP4+ Auto FRR for unicast routes:

- To enable only BGP4+ Auto FRR for unicast routes, run the **auto-frr** command.
- To enable BGP4+ Auto FRR for unicast routes and configure the function of preferentially selecting the SRv6 BE path of the primary route as the backup path, run the **auto-frr best-effort** command.

### Step 5 Run commit

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, check whether the configuration has taken effect.

Run the **display ipv6 routing-table *ipv6-address prefix-length* [ longer-match ] [ verbose ]** command to check the backup forwarding information in the routing table.

### 1.1.10.1.20 Setting a Specified BGP4+ Peer or Each Peer in a Peer Group as an Independent Update Peer-Group

Setting a specified peer or each peer in a peer group as an independent update peer-group prevents routes learned from the peer from being sent back to the peer.

## Usage Scenario

To improve the efficiency of route advertisement, BGP uses the dynamic update peer-group mechanism. The BGP peers with the same configurations are placed in an update peer-group. These routes are grouped once and then sent to all peers in the update peer-group. However, the routes learned from a peer may be sent back to the peer, for example, the preferred route learned from an EBGP peer is sent back to the EBGP peer, or the preferred route that an RR learns from a client is reflected back to the client. In this case, messages are discarded, wasting network resources.

To address this problem, you can set a specified peer or each peer in a peer group as an independent update peer-group so that the routes learned from the peer are not sent back to the peer.

#### NOTE

Setting a specified peer or each peer in a peer group as an independent update peer-group can be performed in multiple address families. This section uses the IPv6 unicast address family as an example. For details about the address families supported, see **peer update-group-independent**.

## Pre-configuration Tasks

Before setting a specified peer or each peer in a peer group as an independent update peer-group, [configure basic BGP4+ functions](#).

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **bgp *as-number***

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

### Step 4 Set a specified peer or each peer in a peer group as an independent update peer-group.

#### NOTE

The configuration of a peer takes precedence over that of the peer group to which the peer belongs.

- To set a specified peer as an independent update peer-group, run the **peer { ipv4-address | ipv6-address } update-group-independent enable** command.
- To set each peer in a peer group as an independent update peer-group, run the **peer group-name update-group-independent** command.

### Step 5 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After setting a specified peer or each peer in a peer group as an independent update peer-group, check the configurations.

Run the **display bgp ipv6 update-peer-group** command to check information about update peer-groups.

### 1.1.10.1.21 Configuring the Route Server Function

This section describes how to configure the route server function. The function reduces network resource consumption on ASBRs.

## Usage Scenario

In some scenarios on the live network, to achieve network traffic interworking, EBGP full-mesh connections may be required. Full-mesh connections between border devices have high requirements on cost and device performance, and are not conducive to the expansion of the network topology and the number of devices. The route server function is similar to the RR function used in IBGP full-mesh connection scenarios. It allows one or more routing devices to advertise routes to their clients (border devices) without changing path attributes, such as AS\_Path, Nexthop, and MED, reducing the consumption of full-mesh connections on each border router.

## Pre-configuration Tasks

Before configuring the route server function, [configure basic BGP4+ functions](#).

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Run **peer { ipv6-address | group-name } route-server-client**

The route server function is enabled on the device, and an EBGP peer is specified as its client.

**Step 5** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring the route server function, verify the configuration.

Run the **display bgp ipv6 routing-table** command to view routes in the BGP4+ routing table.

### 1.1.10.1.22 Configuring the BGP4+ GR Helper

A BGP4+ GR helper helps its neighbor complete BGP4+ GR.

## Usage Scenario

When BGP4+ restarts, peer relationships are re-established and traffic forwarding is interrupted. Enabling Graceful restart (GR) can prevent traffic interruption.

BGP4+ GR needs to be enabled to prevent traffic interruption in the event of BGP4+ restart. A GR restarter and its BGP4+ peer negotiate to establish a GR-capable BGP4+ session.

## Pre-configuration Tasks

Before configuring BGP4+ GR helper, [configure basic BGP4+ functions](#).

## Enabling BGP4+ GR

Enabling or disabling GR may delete and re-establish all sessions and instances.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3 Run graceful-restart**

BGP4+ GR is enabled.

**Step 4 Run commit**

The configuration is committed.

----End

## Configuring BGP4+ GR Session Parameters

You can change the restart time to reestablish a BGP peer relationship.

### Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run bgp *as-number***

The BGP view is displayed.

**Step 3 Run graceful-restart timer wait-for-rib *time***

The period during which the restarting speaker and receiving speaker wait for End-of-RIB messages is set.



You can adjust GR parameter values for the BGP4+ session as needed. However, retaining default parameter values is recommended.

**Step 4 Run commit**

The configuration is committed.

----End

### Verifying the Configuration

After configuring BGP4+ GR helper, check the BGP4+ GR status.

### Prerequisites

The BGP4+ GR helper has been configured.

### Procedure

- Run the **display bgp ipv6 peer verbose** command to check the BGP4+ GR status.

----End

### 1.1.10.1.23 Configuring BMP

The BGP Monitoring Protocol (BMP) monitors BGP4+ running status of devices in real time, such as the establishment and termination status of BGP4+ peer relationships and route update status.

## Usage Scenario

Without BMP, you have to run a query command on a BGP4+ device if you want to learn the BGP4+ running status of the device, which is inconvenient. To improve the network monitoring efficiency, you can configure BMP so that the BGP4+ running status of a client can be monitored by servers.

## Pre-configuration Tasks

Before configuring BMP, [configure basic BGP4+ functions](#).

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **bmp**

BMP is started, and the BMP view is displayed.

### Step 3 (Optional) Run **statistics-timer time**

An interval is set. It determines the frequency at which the BMP device sends BGP4+ running statistics to a monitoring server.

Configure the interval based on BGP4+ stability requirements. In the case of high BGP4+ stability requirements, configure a short interval. However, if the device frequently sends BGP4+ running statistics, a large number of bandwidth resources will be consumed.

### Step 4 Run **bmp-session [ vpn-instance vrf-name ] ipv6-address [ alias alias-name ]**

An IPv6 session address is specified for the TCP connection to be established between the BMP device and the monitoring server.

**alias alias-name** specifies an alias for a BMP session. If the device needs to establish multiple TCP connections with the same monitoring server through different port numbers, specify one IPv6 address and different session aliases (through the **alias alias-name** parameter) for differentiation.

### Step 5 Set the type of route whose statistics are to be sent by the BMP device to the monitoring server.

- Configure the BMP device to send statistics about RIB-in routes (of a specified type) of BGP4+ peers in a specified address family to the monitoring server.

a. Run one of the following commands to enter the BMP-Monitor view:

- monitor public**: displays the BMP-Monitor view and allows the BGP4+ running status of all BGP4+ peers in the public network address family to be monitored.
- monitor all-vpn-instance**: displays the BMP-Monitor view and allows the BGP4+ running status of BGP4+ peers in all VPN instance address families to be monitored.

- **monitor peer**: displays the BMP-Monitor view and allows the BGP4+ running status of a specified BGP4+ peer in the public address family to be monitored.
  - **monitor vpn-instance**: displays the BMP-Monitor view and allows the BGP4+ running status of all BGP4+ peers in a specified VPN instance address family to be monitored.
  - **monitor vpn-instance peer**: displays the BMP-Monitor view and allows the BGP4+ running status of a specified BGP4+ peer in a specified VPN instance address family to be monitored.
- b. Run the **route-mode { ipv6-family unicast | ipv6-family vpng6 } adj-rib-in { pre-policy | post-policy }** command to configure the BMP device to send statistics about RIB-in routes of BGP4+ peers in a specified address family to the monitoring server.

To configure the device to send statistics about all received routes to the monitoring server, specify **pre-policy** in the command. To configure the device to send statistics about only accepted routes to the monitoring server, specify **post-policy** in the command.

 NOTE

If **pre-policy** is specified in the command, run the **keep-all-routes** command in the BGP view to save the routes carried in the BGP4+ Update messages that are received from all BGP4+ peers or peer groups after BGP4+ connections are established, or run the **peer keep-all-routes** command to save the routes carried in the BGP4+ Update messages that are received from a specified BGP4+ peer or peer group after the BGP4+ connection is established.

- Configure the device to send statistics about RIB-out routes of BGP4+ peers in a specified address family to the monitoring server.
    - a. Run one of the following commands to enter the BMP-Monitor view:
      - **monitor public**: displays the BMP-Monitor view and allows the BGP4+ running status of all BGP4+ peers in the public network address family to be monitored.
      - **monitor all-vpn-instance**: displays the BMP-Monitor view and allows the BGP4+ running status of BGP4+ peers in all VPN instance address families to be monitored.
      - **monitor peer**: displays the BMP-Monitor view and allows the BGP4+ running status of a specified BGP4+ peer in the public address family to be monitored.
      - **monitor vpn-instance**: displays the BMP-Monitor view and allows the BGP4+ running status of all BGP4+ peers in a specified VPN instance address family to be monitored.
      - **monitor vpn-instance peer**: displays the BMP-Monitor view and allows the BGP4+ running status of a specified BGP4+ peer in a specified VPN instance address family to be monitored.
- b. Run the **route-mode { ipv6-family unicast | ipv6-family vpng6 } adj-rib-out { pre-policy | post-policy }** command to configure the device to send statistics about RIB-out routes of BGP4+ peers in a specified address family to the monitoring server.

If you want the monitoring server to monitor all the routes to be advertised, regardless of whether they match the export policy, specify **pre-policy** in the command. If you want the monitoring server to monitor only the advertised routes (those that match the export policy), specify **post-policy** in the command.

- Configure the BMP device to send statistics about Local-RIB routes of BGP4+ peers in a specified address family to the monitoring server.
  - a. Run one of the following commands to enter the BMP-Monitor view:
    - **monitor public**: displays the BMP-Monitor view and allows the BGP4+ running status of all BGP4+ peers in the public address family to be monitored.
    - **monitor vpn-instance**: displays the BMP-Monitor view and allows the BGP4+ running status of all BGP4+ peers in a specified VPN instance address family to be monitored.
  - b. Run the **route-mode { ipv6-family unicast | ipv6-family vpng6 } local-rib [ add-path | all ] [ path-marking ]** command to configure the BMP device to send statistics about Local-RIB routes of BGP4+ peers in a specified address family to the monitoring server.

#### Step 6 Run **quit**

The BMP session view is displayed.

#### Step 7 Run **tcp connect port port-number [ password md5 cipher-password | keychain keychain-name ]**

Parameters for the TCP connection to be established between the device and the monitoring server are configured.

##### NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

#### Step 8 (Optional) Run **connect-interface { interface-type interface-number | ipv6-source-address | interface-type interface-number ipv6-source-address }**

The source interface for sending BMP messages is specified.

#### Step 9 Run **commit**

The configuration is committed.

##### NOTE

If a configuration of a BMP session is changed and the new configuration needs to take effect immediately, run the **reset bmp session** command to reset the BMP session.

----End

## Verifying the Configuration

After the configuration is complete, run the following commands to verify the configuration.

- Run the **display bmp session [ vpn-instance vrf-name ] [ ipv6-address [ alias alias-name ] verbose ]** command to check BMP session configurations.
- Run the **display bgp bmp-monitor { all | vpnv6 ipv4-address | ipv6 { ipv4-address | ipv6-address } | vpnv6 vpn-instance vpn-instance-name { ipv4-address | ipv6-address } }** command to check information about all BGP4+ peers or a specified BGP4+ peer monitored through BMP in a specified address family.

### 1.1.10.1.24 Enabling GR for BGP4+ Peers

After graceful restart (GR) is enabled for BGP4+ peers specified on a BGP4+ speaker, the BGP4+ speaker can negotiate the GR capability with these peers and establish BGP4+ connections with them if negotiation succeeds.

#### Usage Scenario

A BGP4+ restart causes re-establishment of all the involved peer relationships, resulting in traffic interruption. Enabling GR globally can prevent traffic interruption. However, this will disconnect all the BGP4+ peer relationships on a device and cause the device to re-negotiate the GR capability with these peers, which affects all the BGP4+-dependent services running on the live network. To prevent this problem, you can enable GR on a BGP4+ speaker for specified BGP4+ peers so that the BGP4+ speaker negotiates the GR capability only with the specified peers. If negotiation succeeds, BGP4+ connections are established between the BGP4+ speaker and specified peers.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bgp as-number**

The BGP view is displayed.

##### Step 3 Run **peer ipv6-address capability-advertise graceful-restart**

GR is enabled for a specified BGP4+ peer, and the device is configured to advertise the GR capability to the specified peer.

##### Step 4 (Optional) Run **peer ipv6-address graceful-restart timer restart time-value**

The maximum duration is set for a specified peer to wait for the local BGP4+ peer relationship to be re-established, the device is configured to advertise the duration to the peer.

#### NOTE

If the specified peer does not support GR, you are advised to run the **peer ipv6-address local-graceful-restart timer restart restart-time** command instead to set the maximum duration for the device to wait for its BGP4+ peer relationship to be reestablished with the specified peer.

##### Step 5 (Optional) Run **peer ipv6-address graceful-restart peer-reset**

The device is enabled to use the GR mode to reset the BGP4+ connection with the specified peer.

Currently, BGP4+ does not support dynamic capability negotiation. Therefore, each time a BGP4+ capability is changed or a new BGP4+ capability is enabled, a BGP4+ speaker tears down the existing sessions with the affected peers and renegotiates BGP4+ capabilities with these peers. For example, a BGP4+ speaker has established a BGP IPv6 unicast peer relationship with a peer, and the IPv6 service is running properly. A change of BGP4+ capability causes the BGP IPv6 unicast peer relationship to be reestablished, affecting the normal running of the IPv6 service. To address this issue, run the **peer ipv6-address graceful-restart peer-reset** command.

**Step 6** (Optional) Run **peer ipv6-address graceful-restart timer wait-for-rib time-value**

The maximum duration is set for the device to wait for the End-of-RIB flag from the specified peer.



If a specified peer does not support GR, run the **peer ipv6-address local-graceful-restart timer wait-for-rib wrftime** command instead to set the maximum duration.

**Step 7** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

- Run the **display bgp ipv6 peer verbose** command to check the BGP4+ GR status.
- Run the **display bgp graceful-restart status** command to check the GR information on the BGP4+ speaker.
- Run the **display bgp local-graceful-restart status** command to check information about local GR on the BGP4+ speaker.

### 1.1.10.1.25 Configuring BGP4+ Route Dampening

Configuring BGP4+ route dampening suppresses unstable BGP4+ routes.

#### Usage Scenario

BGP4+ route dampening suppresses unstable routes and improves network stability. With BGP4+ route dampening, a BGP4+ device does not add any unstable routes to its BGP4+ routing table or advertise them to its BGP4+ peers.

Route instability is mainly reflected by route flapping. When a route flaps, it repeatedly disappears from the routing table and then reappears. BGP4+ is applied to complex networks where routes change frequently. Frequent route flapping consumes bandwidth and CPU resources and even seriously affects network operations. Route dampening prevents the adverse impact of continuous route flapping.

With specified-requirement route dampening, you can use a route-policy to differentiate routes. This allows BGP4+ to apply different route dampening parameters to different routes. You can also configure different route dampening parameters for different nodes of the same route-policy. When route flapping

occurs, BGP4+ can use different route dampening parameters to suppress the routes that match the route-policy. For example, on a network, set a long dampening time for routes with a long mask, and set a short dampening time for routes with a short mask (8-bit mask for example).

## Pre-configuration Tasks

Before configuring BGP4+ route dampening, [configure basic BGP4+ functions](#).

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run bgp as-number

The BGP view is displayed.

### Step 3 Configure BGP4+ route dampening parameters.

- Configure EBGP route dampening parameters.
  - a. Run the **ipv6-family unicast** command to enter the IPv6 unicast address family view, or run the **ipv6-family vpng6** command to enter the BGP-VPNv6 address family view.
  - b. Run the **dampening [ half-life-reach reuse suppress ceiling | route-policy route-policy-name | route-filter route-filter-name ] \* [ update-standard ]** command to set EBGP route dampening parameters.
- Configure IBGP route dampening parameters.
  - a. Run the **ipv6-family unicast** command to enter the IPv6 unicast address family view, or run the **ipv6-family vpng6** command to enter the BGP-VPNv6 address family view.
  - b. Run the **dampening ibgp [ half-life-reach reuse suppress ceiling | route-policy route-policy-name | route-filter route-filter-name ] \* [ update-standard ]** command to set IBGP route dampening parameters.

The value of *suppress* must be greater than that of *reuse* and less than that of *ceiling*.

If the **dampening** command is run with a route-policy referenced, BGP4+ applies the route dampening parameters only to the routes that match the route-policy.

### Step 4 Run commit

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, check whether the configuration has taken effect.

- Run the **display bgp ipv6 routing-table dampened** command to check dampened BGP4+ routes.

- Run the **display bgp ipv6 routing-table dampening parameter** command to check configured BGP4+ route dampening parameters.
- Run the **display bgp ipv6 routing-table flap-info [ regular-expression as-regular-expression | as-path-filter { as-path-filter-number | as-path-filter-name } | network-address [ { mask | mask-length } [ longer-match ] ] ]** command to check route flapping statistics.
- Run the **display bgp vpng6 routing-table dampening parameter** command to check configured BGP VPNv6 route dampening parameters.
- Run the **display bgp vpng6 routing-table dampened** command to check dampened BGP VPNv6 routes.

### 1.1.10.1.26 Configuring Suppression on BGP4+ Peer Flapping

Suppression on BGP4+ peer flapping allows a device to delay the establishment of a BGP4+ peer relationship that flaps continuously.

#### Usage Scenario

BGP4+ peer flapping occurs when BGP4+ peer relationships are disconnected and then immediately re-established in a quick sequence that is repeated. Frequent BGP4+ peer flapping is caused by various factors; for example, a link is unstable, or an interface that carries BGP4+ services is unstable. After a BGP4+ peer relationship is established, the local device and its BGP4+ peer usually exchange all routes in their BGP4+ routing tables with each other. If the BGP4+ peer relationship is disconnected, the local device deletes all the routes learned from the BGP4+ peer. Generally, a large number of BGP4+ routes exist, and in this case, a large number of routes change and a large amount of data is processed during frequent peer flapping. As a result, a large number of resources are consumed, causing high CPU usage. To prevent this issue, a device supports suppression on BGP4+ peer flapping. With this function enabled, the local device suppresses the establishment of the BGP4+ peer relationship if it flaps continuously.

#### Pre-configuration Tasks

Before configuring suppression on BGP4+ peer flapping, complete the following task:

- [1.1.10.1.3 Configuring Basic BGP4+ Functions](#)

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **bgp as-number**

The BGP view is displayed.

##### Step 3 Run **peer peer/pv6Addr oscillation-dampening**

BGP4+ is enabled to suppress the establishment of a specified peer relationship that flaps continuously.

To immediately remove the suppression, you can run the **peer oscillation-dampening disable** command. Alternatively, you can run a reset command or

another command that can cause the peer relationship to be disconnected and re-established.

#### Step 4 Run **commit**

The configuration is committed.

----End

### Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp ipv6 peer verbose** command to check the suppression status of the flapping BGP4+ peer relationship and the remaining time to establish the BGP4+ peer relationship.

#### 1.1.10.1.27 Configuring Flapping Suppression Involved in BGP4+ Next Hop Recursion

Flapping suppression involved in next-hop recursion prevents the system from frequently processing changes in routes that recurse to a frequently flapping next hop, thereby reducing system resource consumption and CPU usage.

### Usage Scenario

If a large number of routes recurse to the same next hop that flaps frequently, the system will be busy processing changes of these routes, which consumes excessive system resources and leads to high CPU usage. To address this problem, configure flapping suppression involved in next-hop recursion.

By default, flapping suppression involved in next-hop recursion is enabled. After flapping suppression involved in BGP4+ next-hop recursion is enabled, a BGP4+ device determines whether to increase, retain, or clear the penalty value by comparing the flapping interval with the configured threshold. When the penalty value exceeds 10, the device suppresses next hop recursion flapping. For example, if the intervals for increasing, retaining, and clearing the penalty value are T1, T2, and T3, respectively, the device calculates the penalty value as follows:

- Increases the penalty value by 1 if the flapping interval is less than T1.
- Retains the penalty value if the flapping interval is greater than or equal to T1, but less than T2.
- Reduces the penalty value by 1 if the flapping interval is greater than or equal to T2, but less than T3.
- Clears the penalty value if the flapping interval is greater than or equal to T3.

### Pre-configuration Tasks

Before configuring flapping suppression involved in BGP4+ next-hop recursion, [configure basic BGP4+ functions](#).

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Run **undo nexthop recursive-lookup restrain disable**

Flapping suppression involved in next-hop recursion is enabled.

If you are not concerned about the system becoming occupied processing route changes and the possible high CPU usage, run the **nexthop recursive-lookup restrain disable** command to disable flapping suppression involved in next-hop recursion.

**Step 5** Run **quit**

Return to the BGP view.

**Step 6** Run **nexthop recursive-lookup restrain suppress-interval add-count-time hold-interval hold-count-time clear-interval clear-count-time**

The intervals are configured for increasing, retaining, and clearing the penalty value for flapping suppression involved in next-hop recursion.

**Step 7** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display bgp ipv6 routing-table** command to check BGP4+ public network route information.
- Run the **display bgp vpng6 routing-table** command to check BGP VPNv6 and BGP VPN route information.

### 1.1.10.1.28 Configuring BGP-LS (IPv6)

BGP-LS (IPv6) provides a simple and efficient method of collecting topology information.

## Usage Scenario

Without BGP-LS, the router uses an IGP (OSPF, OSPFv3, or IS-IS) to collect network topology information and report the topology information of each area to the controller separately. This method has the following disadvantages:

- The controller must have high computing capabilities and support the IGP and its algorithm.
- The controller cannot gain the complete inter-IGP area topology information and therefore is unable to calculate optimal E2E paths.

- Different IGPs report topology information separately to the controller, which complicates the controller's analysis and processing.

With the BGP-LS feature, the topology information discovered by the IGP is summarized by BGP and then sent to the upper-layer controller. With the powerful route selection capability of BGP, BGP-LS has the following advantages:

- Reduces computing capability requirements and spares the necessity of IGPs on the controller.
- Facilitates route selection and calculation on the controller by using BGP to summarize process or AS topology information and report the complete information to the controller.
- Requires only one routing protocol (BGP) to report topology information to the controller.

BGP-LS needs to be deployed on the devices connected to the controller.

## Pre-configuration Tasks

Before configuring BGP-LS, complete the following task:

- **Configure basic IPv6 IS-IS functions or basic OSPF functions or basic OSPFv3 functions.**

## Procedure

1. Enable IGP topology advertisement based on network configurations.
  - Enable IS-IS topology advertisement.
    - i. Run **system-view**  
The system view is displayed.
    - ii. Run **isis [ process-id ]**  
An IS-IS process is configured.
    - iii. Run **cost-style { narrow | wide | wide-compatible | { compatible | narrow-compatible } [ relax-spf-limit ] }**  
A cost style is set for the routes to be accepted and sent by the IS-IS device.
    - iv. Run **traffic-eng [ level-1 | level-2 | level-1-2 ]**  
TE is enabled at a specified level for the IS-IS process.
    - v. Run **ipv6 traffic-eng [ level-1 | level-2 | level-1-2 ]**  
IPv6 TE is enabled at the specified level for the IS-IS process.
    - vi. Run **segment-routing ipv6 locator locator-name [ auto-sid-disable ]**  
IS-IS SRv6 is enabled.
    - vii. Run **ipv6 bgp-ls enable [ level-1 | level-2 | level-1-2 ] [ exclude-prefix ]**  
IS-IS topology advertisement is enabled.
    - viii. (Optional) Run **bgp-ls identifier identifier-value**  
A BGP-LS identifier is configured for IS-IS.

- ix. (Optional) Run **bgp-ls report-exclude { metric-delay-average | metric-delay-variation | link-msd } \***  
Filtering of IS-IS topology information to be advertised is configured.
  - x. Run **commit**  
The configuration is committed.
  - Enable OSPF topology advertisement.
    - i. Run **system-view**  
The system view is displayed.
    - ii. Run **ospf [ process-id | router-id router-id | vpn-instance vpn-instance-name ] \***  
An OSPF process is configured.
    - iii. Run **bgp-ls enable**  
OSPF topology advertisement to BGP is enabled.
    - iv. (Optional) Run **bgp-ls identifier identifier-value**  
A BGP-LS identifier is configured for OSPF.
    - v. (Optional) Run **bgp-ls report-exclude { metric-delay-average | metric-delay-variation } \***  
Filtering of the OSPF topology information to be advertised is configured.
    - vi. Run **commit**  
The configuration is committed.
  - Enable OSPFv3 topology advertisement.
    - i. Run **system-view**  
The system view is displayed.
    - ii. Run **ospfv3 [ process-id ] [ vpn-instance vpn-instance-name ]**  
An OSPFv3 process is created.
    - iii. Run **bgp-ls enable**  
OSPFv3 topology advertisement is enabled.
    - iv. (Optional) Run **bgp-ls identifier identifier-value**  
A BGP-LS identifier is configured for OSPFv3.
    - v. (Optional) Run **bgp-ls report-exclude { metric-delay | metric-delay-average | metric-delay-variation } \***  
Filtering of the OSPFv3 topology information to be advertised is configured.
- After BGP-LS is configured, the device collects OSPFv3 topology information and reports it to the controller. The collected information includes the maximum, minimum, and average delays and delay variation. If you want the device not to report some of the information, you can configure the topology advertisement filtering function.
- vi. Run **commit**  
The configuration is committed.

2. Enable BGP-LS.
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **bgp as-number**  
BGP is enabled, and the BGP view is displayed.
  - c. Run **peer ipv6-address as-number { as-number-plain | as-number-dot }**  
The IP address and AS number of a BGP peer are specified.
  - d. Run **link-state-family unicast**  
BGP-LS is enabled, and the BGP-LS address family view is displayed.
  - e. Run **peer ipv6-address enable**  
BGP-LS route exchange with the specified peer is enabled.
  - f. (Optional) Run **domain identifier domain-id**  
A BGP-LS domain ID is configured.  
A BGP-LS domain ID identifies a device with BGP-LS enabled. If no BGP-LS domain ID is configured, a BGP router ID is used as the BGP-LS domain ID by default. The same BGP-LS domain ID can be configured for multiple devices so that the controller calculates routes based on the combined topology information reported by the devices.
  - g. (Optional) Run **domain as { domain as-plain | domain as-dot }**  
A BGP-LS domain AS number is configured.  
Two devices with different BGP AS numbers must have the same BGP-LS domain AS number configured using the **domain as** command so that the controller can combine the topology information about the two ASs for route calculation.
  - h. (Optional) Run **peer ipv6-address reflect-client**  
An RR is configured, and a client is specified.  
The router on which the **peer reflect-client** command is run functions as the RR, and the specified peer functions as a client.
- i. (Optional) Run **peer ipv6-address route-limit limit [ percentage ] [ alert-only | idle-forever | idle-timeout minutes ]**  
The maximum number of BGP-LS routes that can be accepted from the specified peer is set.  
Generally, a BGP-LS routing table contains a large number of routes. If a large number of routes are received from peers, excessive system resources are consumed. To prevent this problem, run this command to set the maximum number of routes that a BGP device is allowed to accept from a peer. If **idle-forever** is configured, the peer relationship will be interrupted and will not be automatically re-established after the

 NOTE

If the clients of an RR are fully meshed, you can run the **undo reflect between-clients** command to disable route reflection among these clients through the RR to reduce the cost.

If a cluster has multiple RRs configured, you can run the **reflector cluster-id cluster-id** command to configure the same cluster ID for all the RRs. This command is applicable only to RRs.

- i. (Optional) Run **peer ipv6-address route-limit limit [ percentage ] [ alert-only | idle-forever | idle-timeout minutes ]**

The maximum number of BGP-LS routes that can be accepted from the specified peer is set.

Generally, a BGP-LS routing table contains a large number of routes. If a large number of routes are received from peers, excessive system resources are consumed. To prevent this problem, run this command to set the maximum number of routes that a BGP device is allowed to accept from a peer. If **idle-forever** is configured, the peer relationship will be interrupted and will not be automatically re-established after the

number of received routes exceeds the threshold. Therefore, configuring **idle-forever** is not recommended.

- j. (Optional) Run **peer ipv6-address route-policy route-policy-name { import | export }**

A route-policy is specified for the BGP-LS routes to be received from or advertised to a peer.

After a route-policy is created, you can run the **peer route-policy** command to use the route-policy to filter the BGP-LS routes to be received from or advertised to a specified peer. The command configuration ensures that only desired routes are accepted or advertised, which helps manage routes and reduces the BGP-LS routing table size and system resource consumption.

- k. (Optional) Run **peer ipv6-address route-update-interval interval**

An interval at which the device sends Update messages carrying the same route prefix to a specified peer is set.

When BGP-LS routes change, the router sends Update messages to notify its peers. If a route changes frequently, to prevent the router from sending Update messages for every change, run this command to set an interval at which the router sends Update messages carrying the same route prefix to a specified peer.

- l. (Optional) Run **peer ipv6-address allow-as-loop num**

The maximum number of AS number repetitions in the AS\_Path attribute is set for a peer or peer group.

- m. Run **commit**

The configuration is committed.

## Verifying the Configuration

After the configuration is complete, run the following commands to verify the configuration.

- Run the **display bgp link-state unicast peer** command to check information about BGP-LS peers and their status.
- Run the **display bgp link-state unicast routing-table** command to check BGP-LS route information.
- Run the **display bgp link-state unicast routing-table statistics** command to check BGP-LS route statistics.

### 1.1.10.1.29 Configuring BGP SAVNET (IPv6)

After BGP source address validation network (SAVNET) is configured on all devices in an AS, the mapping between the source address and the valid inbound interface of data messages can be generated on each node through source prefix advertisement (SPA) and destination prefix probing (DPP) to filter out source address forgery attack traffic and allow valid traffic to pass through. After BGP SAVNET is configured in the access scenario, a source address validation (SAV) table containing the subnet ingress (UNI interface) can be generated to implement the IP address whitelist function. SAV protection is constructed at the edge of the deployment domain, protecting the customer subnets of the deployment domain from accepting packets with forged source addresses from other subnets.

## Prerequisites

Before configuring BGP SAVNET, complete the following tasks:

- **Configure basic BGP4+ functions.**

## Context

SAV is an important method to eliminate the source address forgery attack, which is one of the most concerned network security threats currently. SAV establishes a mapping between a source address and an inbound interface of a router and checks whether messages from the source address reach the router through this inbound interface. However, existing solutions (such as URPF) have limitations in accuracy, flexibility, and deployability, which limits the effect of defending against source address forgery attacks in practice.

To generate accurate SAV rules, a router deployed with SAV needs to know exactly which interface or interfaces a message carrying a valid source address should arrive at. The BGP SAVNET address family enables all nodes on the entire network to obtain the valid source prefix list of each node through SPA. Then, the BGP SAVNET address family obtains the valid inbound interfaces of valid source prefixes through DPP and generates SAV rules.

After SAVNET is configured on all devices in an AS, the mapping between the source address and the valid inbound interface of data messages can be generated on each node through SPA and DPP to filter out source address forgery attack traffic and allow valid traffic to pass through. After BGP SAVNET is configured in the access scenario, an SAV table containing the subnet ingress (UNI interface) can be generated to implement the IP address whitelist function. SAV protection is constructed at the edge of the deployment domain, protecting the customer subnets of the deployment domain from accepting packets with forged source addresses from other subnets.

## Procedure

### Step 1 BGP SAVNET in an intra-AS scenario:

1. Run the **system-view** command to enter the system view.
2. Run the **bgp *as-number*** command to enable BGP and enter the BGP view.
3. Run the **peer *peerIpv6Addr as-number* *as-number*** command to specify the IP address and AS number of a BGP peer.
4. Run the **ipv6-family savnet** command to enable the IPv6 SAVNET address family and enter the BGP-IPv6-SAVNET address family view.
5. Run the **peer *peerIpv6Addr* enable** command to enable the device to exchange routing information with the specified peer. The address specified here must be the address of the directly connected physical interface. Otherwise, the probing result may be incorrect.
6. (Optional) Run the **peer *ipv6-address* reflect-client** command to configure an RR and specify a client.

The router configured with the **peer reflect-client** command functions as an RR, and the specified peer functions as a client.

 NOTE

If the clients of an RR are fully meshed, you can run the **undo reflect between-clients** command to disable route reflection among these clients through the RR to reduce the cost.

If a cluster has multiple RRs configured, you can run the **reflector cluster-id cluster-id** command to configure the same *cluster-id* for all the RRs. This command is applicable only to RRs.

7. Run the **import-route { direct | static | unr | bgp ebgp | { ospfv3 | isis | ripng } process-id } [ route-policy route-policy-name ]** command to import routes from another protocol.
8. Run the **destination-propagation enable** command to enable IPv6 SAVNET DPP so that the local device can originate DPP messages.
9. Run the **savnet ipv6 forwarding check enable** command to enable IPv6 SAVNET forwarding plane validation globally to guide source address validation during message forwarding.
10. (Optional) Run the **destination-propagation interval** command to set an IPv6 SAVNET DPP interval.
11. (Optional) Run the **sav-rule expire-time time** command to set an aging period for IPv6 SAVNET rules so that dynamically generated rules are deleted when the aging period expires. The aging period of IPv6 SAVNET rules must be greater than or equal to twice the DPP interval configured using the **destination-propagation interval** command.
12. (Optional) Run the **ipv6 savnet static sav-rule ipAddr mask-length interface { interface-name | localIfType localIfNum }** command to configure a static IPv6 SAVNET rule, in which the source prefix of data packets and a valid inbound interface are specified.
13. (Optional) Run the **ipv6 savnet static source-prefix ipAddr mask-length router-id routerId** command to configure a static IPv6 SAVNET source prefix. Static source prefixes are applicable to the scenario where source prefixes need to be manually created due to incomplete route advertisement or route asymmetry.
14. Run the **commit** command to commit the configuration.
15. Run the **quit** command to return to the BGP view.
16. Run the **quit** command to return to the system view.
17. (Optional) Run the **interface interface-type interface-number** command to enter the interface view.
18. (Optional) Run the **savnet complete-mode enable** command to enable the complete validation mode.
19. (Optional) Run the **quit** command to return to the system view.
20. (Optional) Run the **reset bgp savnet ipv6** command to reset a specified IPv6 SAVNET peer connection.

 NOTE

Running this command resets the TCP connection established by BGP and causes peer relationship re-establishment, leading to a short-time peer disconnection. Therefore, exercise caution when running this command.

21. Run the **commit** command to commit the configuration.

**Step 2** BGP SAVNET in the access scenario:

1. Run the **system-view** command to enter the system view.
2. Run the **interface interface-type interface-number** command to enter the interface view.
3. Run the **savnet miig tag tag-value type { single-homed | complete-multi-homed }** command to configure a multi-homed inbound interface group.
4. Run the **quit** command to enter the system view.
5. Run the **bgp as-number** command to enable BGP and enter the BGP view.
6. Run the **peer peerIpv6Addr as-number { as-number-plain | as-number-dot }** command to specify the IP address and AS number of a BGP peer.
7. Run the **ipv6-family savnet** command to enable the IPv6 SAVNET address family and enter the BGP-IPv6-SAVNET address family view.
8. Run the **peer peerIpv6Addr enable** command to enable the device to exchange routing information with the specified peer. The address specified here must be the address of the directly connected physical interface. Otherwise, the probing result may be incorrect.
9. (Optional) Run the **peer ipv6-address reflect-client** command to configure an RR and specify a client.

The router configured with the **peer reflect-client** command functions as an RR, and the specified peer functions as a client.

10. Run the **import-route { direct | static | unr | bgp ebgp | { ospfv3 | isis | ripng } process-id } [ route-policy route-policy-name ]** command to import access subnet routing information and specify a route-policy to add the MIIG tag configured at the subnet ingress to the imported routing information.

 **NOTE**

Run the **apply savnet miig-tag agVal** command in the route-policy view to use the route-policy to set the MIIG tag for the SAVNET.

11. Run the **savnet ipv6 forwarding check enable** command to enable IPv6 SAVNET forwarding plane validation globally to guide source address validation during message forwarding.
12. Run the **commit** command to commit the configuration.

----End

## Follow-up Procedure

After the configuration is complete, run the following commands to verify the configuration.

- Run the **display bgp savnet ipv6 peer** command to check information about BGP IPv6 SAVNET peers and their status.
- Run the **display bgp savnet ipv6 routing-table** command to check SAVNET routing information.
- Run the **display bgp savnet ipv6 sav-rule-table verbose** command to check the SAV-rule-table on the device.
- Run the **display bgp savnet ipv6 source-prefix** command to check the valid source prefixes on the device.

- Run the **display bgp savnet ipv6 source-prefix statistics** command to check SAVNET source prefix statistics.
- Run the **display bgp savnet ipv6 miig** command to check information about IPv6 SAVNET MIIGs.

### 1.1.10.1.30 Configuring BGP4+ Security

To improve BGP4+ network security, you can configure BGP4+ authentication and GTSM on the BGP network.

#### Usage Scenario

You can configure the following functions to improve BGP4+ network security:

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

- MD5 authentication

BGP4+ uses TCP as the transport protocol and considers a packet valid if the source address, destination address, source port, destination port, and TCP sequence number of the packet are correct. However, most parameters in a packet are easily accessible to attackers. To protect BGP4+ against attacks, configure MD5 authentication for TCP connections established between BGP4+ peers.

To prevent the MD5 password set on a BGP4+ peer from being decrypted, update the MD5 password periodically.



MD5 authentication is not recommended if high security is required.

- Keychain authentication

A keychain consists of multiple authentication keys, each of which contains an ID and a password. Each key has a lifecycle, and keys are dynamically selected based on the lifecycle of each key. After a keychain with the same rules is configured on the two ends of a BGP4+ connection, the keychains can dynamically select authentication keys to enhance BGP4+ attack defense.

- TCP-AO authentication

The TCP authentication option (TCP-AO) is used to authenticate received and to-be sent packets during TCP session establishment and data exchange. It supports packet integrity check to prevent TCP replay attacks. TCP-AO authentication improves the security of the TCP connection between BGP peers and is applicable to the network that requires high security.

- BGP4+ GTSM

The GTSM mechanism protects the router by checking whether the TTL value in an IP packet header is within a pre-defined range to enhance the system security.



GTSM supports only unicast addresses. Therefore, configure GTSM on all the routers configured with routing protocols.

- BGP4+ RPKI

Resource Public Key Infrastructure (RPKI) improves BGP4+ security by validating the origin ASs of BGP4+ routes.

## Pre-configuration Tasks

Before configuring BGP4+ security, [configure basic BGP4+ functions](#).

### Configuring BGP4+ Authentication

BGP4+ authentication can be configured to enhance security of BGP networks.

### Usage Scenario

BGP4+ authentication includes MD5, TCP-AO, and keychain authentication.

- MD5 authentication

BGP uses TCP as the transport protocol and considers a packet valid if the source address, destination address, source port, destination port, and TCP sequence number of the packet are correct. However, most parameters in a packet are easily accessible to attackers. To protect BGP against attacks, configure MD5 authentication for TCP connections established between BGP peers.

To prevent an MD5 password configured for a BGP peer from being cracked, change the password periodically.

 NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

- Keychain authentication

A keychain consists of multiple authentication keys, each of which contains an ID and a password. Each key has a lifecycle, and keys are dynamically selected based on the lifecycle of each key. After a keychain with the same rules is configured on the two ends of a BGP connection, the keychains can dynamically select authentication keys to enhance BGP attack defense.

- TCP-AO authentication

The TCP authentication option (TCP-AO) is used to authenticate received and to-be sent packets during TCP session establishment and data exchange. It supports packet integrity check to prevent TCP replay attacks. TCP-AO authentication improves the security of the TCP connection between BGP peers and is applicable to the network that requires high security.

 NOTE

BGP MD5 authentication and BGP keychain authentication are mutually exclusive.

## Pre-configuration Tasks

Before configuring BGP4+ authentication, [configure basic BGP4+ functions](#).

## Procedure

### Step 1 Configure MD5 authentication.

1. Run the **system-view** command to enter the system view.
2. Run the **bgp as-number** command to enter the BGP view.
3. Run the **peer { group-name | ipv6-address } password { cipher cipher-password | simple simple-password }** command to configure an MD5 authentication password.

In BGP4+ MD5 authentication, only the MD5 authentication password is set for the TCP connection, and the authentication is performed by TCP. If authentication fails, no TCP connection can be established.

When setting a password, you can select either of the following input modes:

- **cipher cipher-password**: indicates that a password is set using a ciphertext string.
- **simple simple-password**: indicates that a password is set using a cleartext string.

#### NOTE

- The new password is at least eight characters long and contains at least two of the following types: upper-case letters, lower-case letters, digits, and special characters, except the question mark (?) and space.
- For security purposes, you are advised to configure a password in ciphertext mode. To further improve device security, periodically change the password.

4. Run the **commit** command to commit the configuration.

### Step 2 Configure keychain authentication.

1. Run the **system-view** command to enter the system view.
2. Run the **bgp as-number** command to enter the BGP view.
3. Run the **peer { group-name | ipv6-address } keychain keychain-name** command to configure keychain authentication.

You must configure keychain authentication for TCP-based applications on both BGP peers. Note that encryption algorithms and passwords configured for keychain authentication on both peers must be the same; otherwise, a TCP connection cannot be set up between the BGP peers and BGP messages cannot be exchanged.

The keychain specified by *keychain-name* must exist when you configure BGP4+ keychain authentication; otherwise, the TCP connection cannot be established. For keychain configuration details, see "Keychain Configuration" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Configuration Guide > Security*.

4. Run the **commit** command to commit the configuration.

### Step 3 Configure TCP-AO authentication.

1. Run the **system-view** command to enter the system view.
2. Run the **tcp ao tcpaoname** command to create a TCP-AO and enter the TCP-AO policy view.
3. Run the **binding keychain kcName** command to bind the TCP-AO to a keychain.

 NOTE

Before performing this step, complete configuring basic keychain functions in [Pre-configuration Tasks](#) to create a keychain.

4. Run the **key-id** *keyId* command to create a key ID for the TCP-AO and enter the TCP-AO key ID view.
5. Run the **send-id** *snld* **receive-id** *rcvld* command to configure send-id and receive-id for the Key ID.
6. Run the **quit** command to return to the upper-level view.
7. Run the **quit** command to return to the system view.
8. Run the **bgp** *as-number* command to enter the BGP view.
9. Run the **peer** *ipv6-address* **as-number** *as-number* command to specify the IP address of a peer and the number of the AS where the peer resides.
10. Run the **peer** *ipv6-address* **tcp-ao policy** *tcp-ao-name* command to configure TCP-AO authentication for the TCP connection to be set up between BGP peers. The value of the *tcp-ao-name* parameter must be set to the TCP-AO created in step 2.

 NOTE

For the same peer, the authentication modes TCP-AO, MD5, and keychain are mutually exclusive.

11. Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

```
A peer relationship can be set between two peers that have the same authentication information. Run the display bgp ipv6 peer command to check the peer relationship status.
```

## Configuring BGP4+ GTSM

BGP4+ GTSM must be configured on both peers.

## Usage Scenario

The GTSM prevents attacks through TTL detection. An attacker simulates real BGP4+ packets and sends the packets in a large quantity to the router. After receiving the packets, an interface board of the router directly sends the packets to the BGP4+ module of the control plane if the interface board finds that the packets are sent by the local router, without checking the validity of the packets. The control plane of the router needs to process the "legal" packets. As a result, the system becomes abnormally busy and the CPU usage is high.

The GTSM protects the router by checking whether the TTL value in an IP packet header is within a pre-defined range to enhance the system security.

 NOTE

- The GTSM supports only unicast addresses; therefore, the GTSM must be configured on all the routers configured with routing protocols.

## Pre-configuration Tasks

Before configuring the BGP4+ GTSM, complete the following task:

- **Configuring Basic BGP4+ Functions**

Perform the following steps on both BGP4+ peers:

## Procedure

**Step 1** Configure the basic BGP4+ GTSM functions.

1. Run **system-view**

The system view is displayed.

2. Run **bgp as-number**

The BGP view is displayed.

3. Run **peer { group-name | ipv6-address } valid-ttl-hops [ hops ]**

The BGP4+ GTSM is configured.

The valid TTL range of detected packets is [255 - *hops* + 1, 255]. For example, for an EBGP direct route, the number of hops is 1, that is, the valid TTL value is 255.

### NOTE

- When being configured in the BGP view, the GTSM is also applicable to MP-BGP VPNv4 extensions because they use the same TCP connection.
- The GTSM and EBGP-MAX-HOP functions both affect the TTL values of sent BGP4+ messages and they conflict with each other. Thus, for a peer or a peer group, you can use only either of them.

A BGP4+ router that is enabled with GTSM checks the TTL values in all BGP4+ packets. As required by the actual networking, packets whose TTL values are not within the specified range are discarded. If GTSM is not configured on a BGP4+ router, the received BGP4+ packets are forwarded if the BGP4+ peer configuration is matched. Otherwise, the received BGP4+ packets are discarded. This prevents bogus BGP4+ packets from consuming CPU resources.

4. Run **commit**

The configuration is committed.

**Step 2** Set the default action for packets that do not match the GTSM policy.

GTSM only checks the TTL values of packets that match the GTSM policy. Packets that do not match the GTSM policy can be allowed or dropped. If "drop" is set as the default GTSM action for packets, you need to configure TTL values for all the packets sent from valid peers in the GTSM policy. If TTL values are not configured for the packets sent from a peer, the device will discard the packets sent from the peer and cannot establish a connection to the peer. Therefore, GTSM enhances security but reduces the ease of use.

You can enable the log function to record packet drop for troubleshooting.

Perform the following configurations on the GTSM-enabled router:

1. Run **system-view**

The system view is displayed.

2. Run **gtsm default-action { drop | pass }**

The default action for packets that do not match the GTSM policy is configured.

 **NOTE**

If the default action is configured but no GTSM policy is configured, GTSM does not take effect.

This command is supported only on the Admin-VS and cannot be configured in other VSs. This command takes effect on all VSs.

3. Run **commit**

The configuration is committed.

----End

## Checking the Configurations

Run the following command to check the previous configurations.

- Run the **display gtsm statistics { slot-id | all }** command to check the statistics about the GTSM.

 **NOTE**

In VS mode, this command is supported only by the admin VS.

## Configuring ROA

Resource Public Key Infrastructure (RPKI) can be configured to validate the origin of BGP4+ routes to ensure BGP4+ security. Route Origin Authorization (ROA) uses RPKI to validate and filter routes.

## Usage Scenario

To solve the problem of BGP4+ route hijacking, the industry proposes the RPKI solution that validates the origin ASs of BGP4+ routes. Distributed RPKI servers are used to collect information such as the origin AS numbers, route prefixes, and masks of BGP4+ routes initiated by each ISP. After a device sets up a connection with an RPKI cache server, the device saves a copy of Route Origin Authorization (ROA) data locally. If no RPKI server is available, static ROA data can be configured on a device. Inbound ROA validation, which applies to the BGP4+ routes received from peers, can be configured to control route selection. In addition, outbound ROA validation, which applies to the BGP4+ routes to be advertised to peers, can be configured to control route advertisement. ROA validation ensures that hosts in an AS can securely access external services.

## Pre-configuration Tasks

Before configuring ROA, complete the following tasks:

- [Configure basic BGP4+ functions.](#)

- Run the **active port-peering slot slot-id card card-id port port-list** command to activate an interface-specific peering scenario license.

## Procedure

**Step 1** Select one of the following configurations based on the usage scenario:

- If the local device needs to obtain the ROA database through a connection to be established with an RPKI server, perform the following operations:
  - Run **system-view**  
The system view is displayed.
  - Run **rpkı**  
RPKI is started, and the RPKI view is displayed.
  - Run **session ipv6-address**  
An IPv6 address is specified for a TCP connection to be set up between the device and the RPKI server.
  - Run **tcp port port-number [ password md5 cipher-password | keychain keychain-name ]**  
Parameters are configured for the TCP connection to be set up between the device and the RPKI server.

### NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

The password must be at least eight characters long and contain at least two of the following character types: uppercase letters, lowercase letters, digits, and special characters (excluding question marks and spaces).

For security purposes, you are advised to configure a ciphertext password, and change it periodically.

- (Optional) Run **timer { aging aging-time | refresh refresh-time }**

RPKI session timers are configured.

The *aging-time* parameter specifies a value of the validation data age timer, and the *refresh-time* parameter specifies a value of the session update timer. You can configure the timers based on actual requirements on BGP4+ security. Small values are recommended if high BGP4+ security is required. However, frequent data updates consume much network bandwidth.

- (Optional) Run **rpkı-limit limit [ alert-only | idle-forever | idle-timeout times ]**

The maximum number of ROA entries that the device is allowed to accept in a session is configured.

In most cases, a large number of ROA entries exist on a server. If the device receives a large number of ROA entries from the server, excessive system resources will be consumed. To prevent this problem, run the **rpkı-limit** command to configure the maximum number of ROA entries that the BGP4+ device is allowed to accept in a session.

- g. (Optional) Run **connect-interface** { *interface-name* | *ipv6-source-address* | *interface-type interface-number* | *interface-type ipv6-source-address* | *interface-type interface-number ipv6-source-address* }  
The source interface for sending RPKI packets is specified.
- h. (Optional) Run **ssl-policy** *policy-name*  
An SSL policy to be bound to the TCP connection between the device and RPKI server is configured.
- i. Run **quit**  
Exit the RPKI session view.
- j. Run **quit**  
Exit the RPKI view.
- k. Run **commit**  
The configuration is committed.

 **NOTE**

If configurations of an RPKI session are changed and you want its new configurations to take effect immediately, run the **reset rpk session** command to reset the RPKI session.

- If a static ROA database needs to be configured on the local device, perform the following operations:
  - a. Run **system-view**  
The system view is displayed.
  - b. Run **rpk**  
RPKI is started, and the RPKI view is displayed.
  - c. Run **origin-validation**  
A static ROA database is created, and the RPKI origin-validation view is displayed.
  - d. Run **static record** *ipv6-address* *ipv6-mask-length* **max-length** *ipv6-max-mask-length* **origin-as** *as-number*  
A record is configured for the static ROA database.
  - e. Run **quit**  
The RPKI view is displayed.
  - f. Run **quit**  
The system view is displayed.
  - g. Run **commit**  
The configuration is committed.

**Step 2** Run **bgp** *as-number*

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 4** Configure inbound or outbound ROA validation as required.

- To configure inbound ROA validation (validation results not affecting route acceptance) for the routes received from an EBGP peer, perform the following operations:
  - a. Run **prefix origin-validation enable**

Origin AS validation of RPKI is enabled. After origin AS validation is enabled, the device matches the origin AS of each received route against the origin AS data in the database and provides the validation result, which can be Valid, Not Found, or Invalid.
  - b. (Optional) Run **bestroute origin-as-validation [ allow-invalid ]**

The device is configured to apply origin AS validation results of RPKI to BGP4+ route selection.

BGP4+ selects routes in descending order of Valid, Not Found, and Invalid after origin AS validation results are applied to route selection. If **allow-invalid** is not specified in the command, the BGP4+ routes with the validation result being Invalid do not participate in route selection.
  - c. (Optional) Run **peer { ipv6-address | group-name } advertise-ext-community**

The device is configured to advertise extended community attributes to the specified peer.
  - d. (Optional) Run **peer { ipv6-address | group-name } advertise origin-as-validation**

The device is enabled to advertise BGP4+ origin AS validation results of RPKI to the specified peer or peer group.

 **NOTE**

BGP4+ origin AS validation results of RPKI can be advertised only to IBGP peers.

- To configure outbound ROA validation for the routes to be advertised to an EBGP peer to control route advertisement, perform the following operations:

Run **peer { peer/pv6Addr | peerGroupName } origin-validation export [ include-not-found [ external ] ]**

The local device is configured to perform outbound ROA validation on the routes to be advertised to the specified EBGP peer.

After the local device is configured to perform outbound ROA validation on the routes to be advertised to a specified EBGP peer, the device matches the origin ASs of the routes against those of the matched routes recorded in the database. The validation result can be Valid, Not Found, or Invalid. By default, only the routes whose validation result is Valid are advertised. To configure the device to advertise the routes with the validation result being Valid or Not Found, specify the **include-not-found** keyword in the preceding command. To configure the device to advertise the routes with the validation result being Valid or Not Found (if the routes with the result being Not Found were received from another AS), specify the **include-not-found external** keyword in the preceding command.

**Step 5 Run commit**

The configuration is committed.

**----End**

## Verifying the Configuration

After the configuration is complete, verify it.

- Run the **display rpk session *ipv6-address* verbose** command to check RPKI session configurations.
- Run the **display rpk table** command to check ROA information.

## Configuring ASPA

RPKlv2-based ASPA validation verifies the AS\_Path attribute of routes, ensuring BGP4+ security.

## Usage Scenario

RPKlv0-based ROA validation can detect unexpected path leaks, but it relies on the origin AS in the BGP attribute AS\_Path, which may be manipulated by attackers. As an inter-AS routing security mechanism, RPKlv0 provides only origin validation but not path validation.

ASPA can automatically detect invalid AS\_Paths in routes received from peers based on the customer-to-provider shared signature database constructed through RPKlv2.

## Pre-configuration Tasks

Before configuring ASPA, complete the following task:

- [Configure basic BGP4+ functions.](#)

## Procedure

**Step 1** Select one of the following configurations based on the usage scenario:

- If the local device needs to obtain data in the ASPA database through a connection to be established with an RPKI server, perform the following operations:
  - a. Run the **system-view** command to enter the system view.
  - b. Run the **rpk** command to start RPKI and enter the RPKI view.
  - c. Run the **session *ipv6-address*** command to configure session information for the TCP connection between the local device and the RPKI server.
  - d. Run the **tcp port *port-number* [ password *md5 cipher-password* | | keychain *keychain-name* ]** command to configure parameters for the TCP connection between the local device and the RPKI server.

 NOTE

For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use one of the algorithms, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

The MD5 algorithm is not recommended if high security is required.

The password must be at least eight characters long and contain at least two of the following character types: uppercase letters, lowercase letters, digits, and special characters (excluding question marks and spaces).

For security purposes, you are advised to configure a ciphertext password, and change it periodically.

- e. Run the **version version-num** command to configure an RPKI version. RPKIv2 allows a device to receive ASPA data on the condition that both the device and the RPKI server support RPKIv2.
- f. (Optional) Run the **timer { aging aging-time | refresh refresh-time }** command to configure timers for the RPKI session.

The *aging-time* parameter specifies a value of the validation data aging timer, and the *refresh-time* parameter specifies a value of the session update timer. You can configure the timers based on actual requirements on BGP4+ security. Small values are recommended if high BGP4+ security is required. However, frequent data updates consume much network bandwidth.

- g. (Optional) Run the **aspa-limit limit [ percentage ] [ alert-only | idle-forever | idle-timeout times ]** command to configure the maximum number of ASPA pairs that the device is allowed to accept in the session. In most cases, a large number of ASPA pairs exist on a server. If a BGP4+ device receives a large number of ASPA pairs from the server, excessive system resources will be consumed. To prevent this problem, run the **aspa-limit** command to configure the maximum number of ASPA pairs that the device is allowed to accept in the session.
- h. (Optional) Run the **connect-interface { interface-name | ipv4-source-address | interface-type interface-number | interface-type ipv4-source-address | interface-type interface-number ipv4-source-address }** command to specify the source interface for sending RPKI messages.
- i. (Optional) Run the **ssl-policy policy-name** command to configure an SSL policy to be bound to the TCP connection between the device and the RPKI server.
- j. Run the **quit** command to enter the RPKI view.
- k. Run the **quit** command to enter the system view.
- l. Run the **commit** command to commit the configuration.

 NOTE

If configurations of an RPKI session are changed and you want its new configurations to take effect immediately, run the **reset rpk session** command to reset the RPKI session.

- If a static ASPA database needs to be configured on the local device, perform the following operations:
  - a. Run the **system-view** command to enter the system view.

- b. Run the **rpxi** command to start RPKI and enter the RPKI view.
- c. Run the **asp-validation** command to create a static ASPA database and enter the RPKI ASPA-validation view.
- d. Run the **static record customer-as provider as-number { ipv4 | ipv6 }** command to configure the static ASPA database.
- e. Run the **quit** command to enter the RPKI view.
- f. Run the **quit** command to enter the system view.
- g. Run the **commit** command to commit the configuration.

**Step 2** Run the **bgp as-number** command to enter the BGP view.

**Step 3** Configure inbound ASPA validation as required. To configure inbound ASPA validation (validation results not affecting route acceptance) for the routes received from an EBGP peer, perform the following operations:

1. Run the **peer { peerIpv4Addr | peerIpv6Addr } role { provider | rs | rs-client | customer | lateral-peer | sibling }** command to configure a role for a BGP4+ peer.
2. Run the **ipv6-family unicast** command to enter the IPv6 unicast address family view.
3. Run the **asp-validation enable** command to enable RPKI-based ASPA validation. After ASPA validation is enabled, the device compares the AS\_Path of a route with the matching ASPA pair recorded in the database and provides the validation results: Valid, NotFound, or Invalid.
4. (Optional) Run the **bestroute aspa-validation [ allow-invalid ]** command to configure the device to apply ASPA validation results of RPKI to BGP4+ route selection. BGP selects routes in descending order of Valid, Not Found, and Invalid after ASPA validation results are applied to route selection. If **allow-invalid** is not specified in the command, the BGP routes with the validation result being Invalid do not participate in route selection.

**Step 4** Run the **commit** command to commit the configuration.

----End

## Verifying the Configuration

After the configuration is complete, verify it.

Run the **display rpxi aspa ipv6 table** command to check ASPA-related data.

## Configuring Regional Validation

Resource Public Key Infrastructure (RPKI) regional validation or regional confederation validation ensures BGP4+ security by validating route advertisers.

## Usage Scenario

Regional validation: Users can manually combine multiple trusted ASs into a region and combine multiple regions into a regional confederation. Regional validation controls route selection results by checking whether the routes received from EBGP peers in an external region belong to the local region. This prevents intra-region routes from being hijacked by attackers outside the local region, and ensures that hosts in the local region can securely access internal services.

Regional validation applies to the following typical scenarios: regional validation scenario and regional confederation validation scenario.

## Pre-configuration Tasks

Before configuring regional validation, complete the following tasks:

- **Configure basic BGP4+ functions.**
- Run the **active port-peering slot slot-id card card-id port port-list** command to activate an interface-specific peering scenario license.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run rPKI

RPKI is started, and the RPKI view is displayed.

### Step 3 Run region-validation

Regional validation is enabled, and the region-validation view is displayed.

### Step 4 You can configure regions or regional confederation as required.

- Create a region.
  - a. Run **region region-id**  
A region is created.
  - b. Run **description description-text**  
A description is configured for the region.
  - c. Run **as-number { asn } &<1-100>**  
An AS number list is configured so that the AS numbers in it can be added to the region.
  - d. Run **quit**  
Exit the RPKI region-validation-region view.
  - e. Run **quit**  
The system view is displayed.
- Create a regional confederation.
  - a. Run **region region-id**  
A region is created.
  - b. Run **quit**  
Exit the RPKI region-validation-region view.
  - c. Run **region-confederation region-confederation-id**  
A regional confederation is created.
  - d. Run **description description-text**  
A description is configured for the regional confederation.
  - e. Run **region { region-id } &<1-100>**

A region ID list is configured in the regional confederation so that regions in the list are added to the regional confederation.

f. Run **quit**

Exit the RPKI region-validation-confederation view.

g. Run **quit**

The system view is displayed.

**Step 5** Run **bgp as-number**

The BGP view is displayed.

**Step 6** Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

**Step 7** Enable the region or regional confederation function as required.

- Run **region-validation**

BGP4+ regional validation is enabled.

- Run **region-validation confed-check strict**

Strict BGP4+ regional validation is enabled.

**Step 8** Run **bestroute region-validation [ allow-invalid ]**

The device is configured to apply the BGP4+ regional validation results of RPKI to BGP4+ route selection.

If regional validation succeeds, the route is valid and can participate in route selection. If regional validation fails, the route is invalid and cannot participate in route selection. To allow the routes that fail regional validation to be valid and participate in route selection, configure the **allow-invalid** parameter in the command. The priority of such routes is reduced during route selection.

**Step 9** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the **display rpk session ipv6-address verbose** command to check RPKI session configurations.

### 1.1.10.1.31 Configuring BGP4+ 6PE

BGP4+ 6PE enables separated IPv6 networks to communicate using the MPLS tunneling technology.

## Usage Scenario

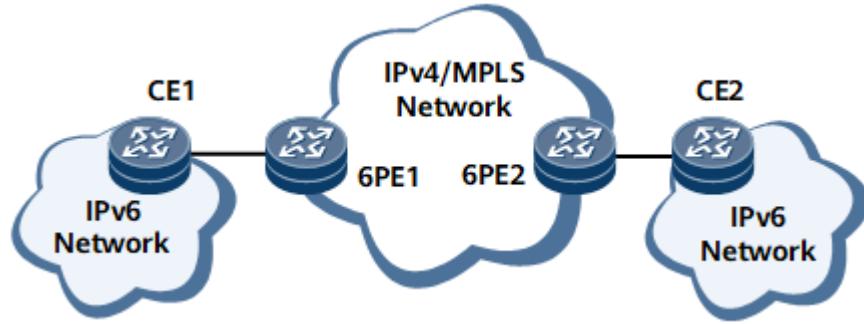
6PE enables IPv6 networks separated by IPv4/MPLS networks to communicate.

Separated IPv6 networks can be connected using various tunneling techniques. A 6PE tunnel is established on Internet Service Provider's (ISP's) PEs that support the IPv4/IPv6 dual stack. The 6PE tunnel identifies IPv6 routes by label assigned by the

Multiprotocol Border Gateway Protocol (MP-BGP), and implements IPv6 forwarding using LSPs between PEs.

As shown in [Figure 1-457](#), the IPv6 network where CE1 and CE2 reside are separated by an IPv4/MPLS network. Configuring 6PE enables CE1 and CE2 to communicate across the IPv4 network.

**Figure 1-457** Networking with 6PE



## Pre-configuration Tasks

Before configuring BGP4+ 6PE, complete the following tasks:

- Connect interfaces and setting parameters for the interfaces to ensure that the physical-layer status of the interfaces is Up.
- Configure the link layer protocol parameters for interfaces.
- Ensure that routes on the IPv4/MPLS backbone network are reachable.

## Configuring the IPv4/IPv6 Dual Stack

The IPv4/IPv6 dual stack needs to be configured on the router at the edge of an IPv6 network and an IPv4 network.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 3 Run **ip address ip-address { mask | mask-length }**

An IPv4 address is configured for the interface.

#### Step 4 Run **quit**

Return to the system view.

#### Step 5 Run **interface interface-type interface-number**

The interface view is displayed.

#### Step 6 Run **ipv6 enable**

IPv6 is enabled on the interface.

- Step 7** Run **ipv6 address { ipv6-address prefix-length | ipv6-address/prefix-length }**  
**eui-64 or ipv6 address { ipv6-address prefix-length | ipv6-address/prefix-length }**
- An IPv6 address is configured for the interface.

- Step 8** Run **commit**

The configuration is committed.

----End

## Configuring an LDP LSP on an IPv4 Network

An LDP LSP is configured on an IPv4/MPLS backbone network to forward IPv6 packets.

## Procedure

- Step 1** Run **system-view**

The system view is displayed.

- Step 2** Run **mpls lsr-id lsr-id**

An LSR ID is configured.

- Step 3** Run **mpls**

MPLS is enabled and the MPLS view is displayed.

- Step 4** (Optional) Run **lsp-trigger { all | host | ip-prefix ip-prefix-name | none }**

An LSP triggering policy is configured.

Currently, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X automatically distributes labels to host routes with 32-bit masks. To distribute labels to routes of other types or specific routes, run this command.

- Step 5** Run **quit**

Return to the system view.

- Step 6** Run **mpls ldp**

LDP is enabled.

- Step 7** Run **quit**

Return to the system view.

- Step 8** Run **interface interface-type interface-number**

The interface view is displayed.

- Step 9** Run **mpls**

MPLS is enabled on the interface.

- Step 10** Run **mpls ldp**

LDP is enabled on the interface.

**Step 11 Run commit**

The configuration is committed.

----End

## Establishing a 6PE Peer Relationship Between PEs

6PE peers can exchange IPv6 routes learned from their attached CEs.

### Procedure

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run bgp *as-number***

The BGP view is displayed.

**Step 3 Run peer *ipv4-address as-number as-number***

The IP address of the peer and the number of the AS where the peer resides are specified.

**Step 4 Run peer *ipv4-address connect-interface interface-type interface-number***

The interface that is used to establish a connection with the remote PE is specified.

**Step 5 Run ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

**Step 6 Run peer *ipv4-address enable***

A 6PE peer relationship is established.

**Step 7 Run peer *peer/pv4Addr label-route-capability-only***

The 6PE labeled route exchange capability is enabled.

In the 6PE scenario, after the **peer label-route-capability** command is run, peers negotiate the label and unicast capabilities and advertise labeled routes only to the specified peer. However, in the 6PE scenario, peers do not need to negotiate the unicast capability. If the **peer label-route-capability-only** command is run, the device negotiates only the label capability and advertises labeled routes only to the specified peer. Therefore, the **peer label-route-capability-only** command is recommended in the 6PE scenario.

**Step 8 Run commit**

The configuration is committed.

**Step 9 Run quit**

Exit the BGP-IPv6 unicast address family view.

**Step 10 Run quit**

Exit the BGP view.

**Step 11** (Optional) Run **mpls 6pe ttl-mode { pipe | uniform }**

A mode of processing MPLS TTLs in 6PE route labels is configured.

**Step 12** Run **commit**

The configuration is committed.

----End

### (Optional) Enabling 6PE Routes Sharing the Explicit Null Label

By enabling IPv6 provider edge (6PE) routes sharing the explicit null label, you can save label resources on 6PE devices.

#### Context

By default, the 6PE device applies for a label for each 6PE route. When a large number of 6PE routes need to be sent, a large number of labels are required. This greatly wastes label resources and causes IPv6 routes unable to be advertised due to the shortage of label resources.

After 6PE routes sharing the explicit null label is enabled, all 6PE routes to be sent to the same 6PE peer share the explicit null label 2.

#### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **bgp as-number**

The BGP view is displayed.

**Step 3** Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

**Step 4** Run **apply-label explicit-null**

All 6PE routes to be sent to the same 6PE peer share the explicit null label.

If you run this command after a 6PE peer relationship is established, temporary packet loss occurs.

**Step 5** Run **commit**

The configuration is committed.

----End

### Configuring Route Exchange Between a PE and a CE

An IPv6 routing protocol needs to be configured on a PE and a CE to enable them to learn IPv6 routes from each other.

## Context

The routing protocol running between a PE and a CE can be EBGP, IBGP, IPv6 static route, OSPFv3, RIPv2, or IS-IS. You can select one of them as required. For details, see the configuration of each routing protocol in the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide - IP Routing*.

## Verifying the Configuration

After configuring BGP4+ 6PE, check whether CEs can learn routes to each other.

## Procedure

- Step 1** Run the **display bgp ipv6 peer** command on each PE to check the BGP peer relationship status.
  - Step 2** Run the **display mpls ldp session vpn-instance vpn-instance-name [ peer-id | verbose ]** command on PEs to check the status of the LDP session between the PEs.
  - Step 3** Run the **display bgp ipv6 routing-table ipv6-address prefix-length** command on each PE or the **display ipv6 routing-table ipv6-address prefix-length [ longer-match ] [ verbose ]** command on each CE to check the routes destined for the remote IPv6 network.
- End

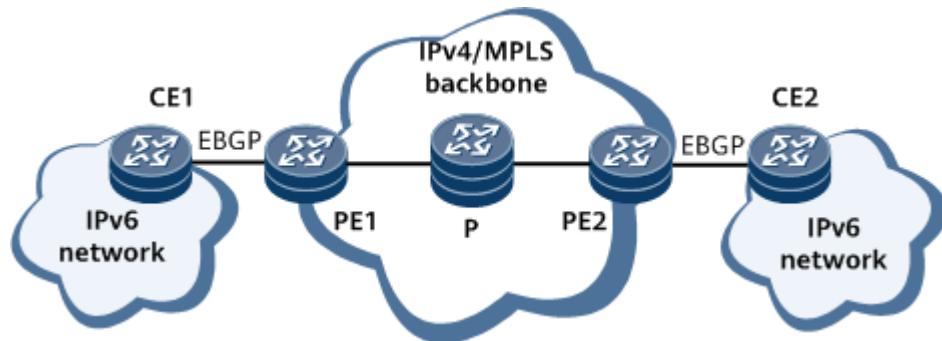
### 1.1.10.1.32 Enabling the Device to Recurse BGP IPv6 Unicast Routes to LSPs

By default, BGP IPv6 unicast routes can recurse to outbound interfaces and next hops, but not to LSPs. A device can be enabled to recurse BGP IPv6 unicast routes to LSPs.

## Usage Scenario

On the IPv4/MPLS backbone network shown in [Figure 1-458](#), PE1, P, and PE2 run an IGP for interworking, an LDP LSP or MPLS TE tunnel is established between PE1 and PE2, and a public network IPv6 EBGP peer relationship is established between CE1 on an IPv6 network and PE1 and between CE2 on another IPv6 network and PE2. To allow PE1 and PE2 to exchange BGP IPv6 unicast routes, an IBGP peer relationship is established between PE1 and PE2 using loopback interfaces, and a BGP IPv4 peer relationship is established between PE1 and PE2 in the IPv6 unicast address family. CE1 sends an IPv6 route to PE1 through the EBGP peer connection. Upon receipt of the route, PE1 changes the next-hop IP address of the route to a local IP address and sends the route to PE2 through the IBGP peer connection. After PE2 learns the IPv6 route, it can recurse the route only to an outbound interface and next hop by default. However, PE2 can be enabled to recurse BGP IPv6 unicast routes to LSPs.

Figure 1-458 Scenario where BGP IPv6 unicast routes recurse to LSPs



## Pre-configuration Tasks

Before enabling the device to recurse BGP IPv6 unicast routes to LSPs, complete the following tasks:

- Configure an IGP.
- Establish an LDP LSP or MPLS TE tunnel between PE1 and PE2.
- Establish an IBGP peer relationship between PE1 and PE2 using loopback interfaces, and establish a BGP IPv4 peer relationship between PE1 and PE2 in the IPv6 unicast address family view.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **bgp as-number**

The BGP view is displayed.

### Step 3 Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

### Step 4 Run **unicast-route recursive-lookup tunnel-v4 [ tunnel-selector tunnel-selector-name ]**

The device is enabled to recurse BGP IPv6 unicast routes to tunnels.

To enable the device to recurse BGP IPv6 unicast routes to MPLS TE tunnels, specify **tunnel-selector tunnel-selector-name** in the command.

### Step 5 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After the device is enabled to recurse BGP IPv6 unicast routes to LSPs, you can run the **display bgp ipv6 routing-table ipv6-address** command to view information about such route recursion.

### 1.1.10.1.33 Maintaining BGP4+

Maintaining BGP4+ involves resetting BGP4+ connections, clearing BGP4+ statistics, and debugging BGP4+.

#### Resetting BGP4+ Connections

Resetting BGP4+ connections interrupts peer relationships.

#### Context

##### NOTICE

The BGP peer relationship between routers is torn down if the **reset bgp ipv6** command is run to reset the BGP4+ connection. Exercise caution when resetting BGP connections.

When a BGP4+ routing policy (the router does not support the route-refresh capability) is changed, reset BGP connections so that the new configuration can take effect. To reset BGP4+ connections, run any of the following commands in the user view:

#### Procedure

- Run the **reset bgp ipv6 all** command to reset all BGP4+ connections.
- Run the **reset bgp ipv6 as-number** command to reset a BGP4+ connection between two devices in different ASs.
- Run the **reset bgp ipv6 { ipv4-address | ipv6-address }** command in the user view to reset the BGP4+ connection with a specified peer.
- Run the **reset bgp ipv6 external** command in the user view to reset all external BGP4+ connections.
- To reset all internal BGP4+ connections, run the **reset bgp ipv6 internal** command in the user view.
- To reset the BGP4+ connection with a specified slow peer, run the **reset bgp ipv6 [ ipv6-address ] slow-peer** command in the user view.

----End

#### Clearing BGP4+ Statistics

Clearing BGP4+ statistics involves clearing route flapping statistics and route dampening statistics.

#### Context

##### NOTICE

BGP4+ statistics cannot be restored after being cleared. Therefore, exercise caution when performing this operation.

## Procedure

- To clear route flapping statistics, run the **reset bgp ipv6 flap-info [ regexp as-path-regexp | as-path-filter { as-path-filter-number | as-path-filter-name } | ipv6-address prefix-length ]** command in the user view.
- To clear route flapping statistics of a specified peer, run the **reset bgp ipv6 ipv6-address flap-info** command in the user view.
- To clear route dampening information and release suppressed routes, run the **reset bgp ipv6 dampening [ ipv6-address prefix-length ]** command in the user view.

----End

## Clearing the SIDs That Are in the Delayed Deletion State in BGP4+

SIDs in the delayed deletion state consume SID resources. If SID resources are insufficient, other services may fail to apply for SIDs. In this case, you can clear the SIDs in the delayed deletion state from BGP4+.

## Procedure

- To immediately clear all SIDs in the delayed deletion state from BGP4+, run the **reset bgp slow-delete sid ipv6** command in the user view.

### NOTE

Clearing the SIDs in the delayed deletion state may cause traffic forwarding failures. Therefore, exercise caution when performing this operation.

- To immediately clear the SIDs in the delayed deletion state from the BGP-VPN instance IPv6 address family, run the **reset bgp slow-delete sid ipv6 vpn** command in the user view.

----End

## Configuring a Mode in Which BGP4+ Path Attributes Are Processed

To enhance reliability, you can configure a special mode in which BGP4+ path attributes are processed.

## Usage Scenario

A BGP4+ Update message contains various path attributes. If a local device receives Update messages containing malformed path attributes, the involved BGP4+ sessions may flap. To resolve this issue and enhance reliability, configure a special mode in which a device processes specified BGP4+ path attributes in received Update messages. Special modes indicate those that are not defined in a standard protocol.

### NOTE

This task may cause path attribute discarding and route withdrawal. Therefore, exercise caution when performing this task.

This function takes effect immediately for the routes received after the **bgp path-attribute** or **peer path-attribute-treat** command is run. However, this function does not take effect immediately for the routes received before the **bgp path-attribute** or **peer path-attribute-treat** command is run. To allow this function to take effect in this case, you need to run the **refresh bgp** command.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Configure a way to handle incorrect path attributes as required.

- To allow the device to accept the path attributes with the value of 0, run the **bgp path-attribute { originator-id | attr-set } accept-zero-value** command.
- To allow the device to accept the path attributes with the length of 0, run the **bgp path-attribute { community | ext-community | ipv6-ext-community | large-community | attr-set | wide-community | clust-list } accept-zero-length** command.

The **bgp path-attribute accept-zero-value** command allows the path attributes with the value of 0 to be accepted. The **bgp path-attribute accept-zero-length** command allows the path attributes with the length of 0 to be accepted.

After the **bgp path-attribute attr-set accept-zero-value** command is run, if the Originator\_ID in the Attr\_Set attribute is 0, the corresponding route is accepted. After the **bgp path-attribute attr-set accept-zero-length** command is run, if the length of the Community, Ext-community, IPv6 ext-community, Large-community, Wide-community, or Cluster\_List attribute in the Attr\_Set attribute is 0, the corresponding route is accepted.

#### NOTE

The **bgp path-attribute** command takes effect for all address families. To configure the device to perform special processing on path attributes in a specific address family, perform the following steps.

The **bgp path-attribute accept-zero-value** command takes effect for all address families. To configure the device to perform special processing on path attributes in a specific address family, run the **peer peer/pv4Addr treat-with-error attribute-id id accept-zero-value** command. Currently, the **attribute-id id** supports only the Originator\_ID attribute.

The **bgp path-attribute wide-community accept-zero-length** command does not take effect in the BGP RPD address family. After a device receives an RPD route with the Wide-Community attribute whose length is 0, the device still withdraws the route even if this command is run.

### Step 3 Run **bgp as-number**

BGP4+ is started (with the local AS number specified), and the BGP view is displayed.

### Step 4 Run **peer { ipv4-address | ipv6-address } as-number as-number**

The IP address of a peer and the number of the AS where the peer resides are specified.

### Step 5 Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

### Step 6 Run **peer { peer/pv4Addr | peer/pv6Addr } path-attribute-treat attribute-id { id [ to id2 ] } &<1-255> { discard | withdraw | treat-as-unknown }**

A mode for the device to process specified attributes is configured.

 NOTE

Running this command may cause path attribute discarding and route withdrawal.  
Therefore, exercise caution when running this command.

If both the **bgp path-attribute** and **peer path-attribute-treat attribute-id** commands are configured, the device executes the **peer path-attribute-treat attribute-id** command.

The **peer path-attribute-treat** parameter specifies a path attribute processing mode, which can be any of the following ones:

- Discarding specified attributes
- Withdrawing the routes with specified attributes
- Processing specified attributes as unknown attributes

**Step 7** Run **commit**

The configuration is committed.

----End

## Configuring Whitelist Session-CAR for BGP4+

You can configure whitelist session-CAR for BGP4+ to isolate bandwidth resources by session for BGP4+ messages. This configuration prevents bandwidth preemption among BGP4+ sessions in the case of a traffic burst.

### Context

The function of whitelist session-CAR for BGP4+ sets an independent CAR channel for each BGP4+ session to ensure that the bandwidth of each BGP4+ session is not preempted by other traffic (including traffic from other sessions of the same protocol and traffic from other protocols). When BGP4+ messages suffer a traffic burst, you can configure this task to adjust the message channel bandwidth of each BGP4+ session in the BGP4+ whitelist session CAR to ensure that BGP4+ messages are sent to the CPU normally.

 NOTE

If the function becomes abnormal or affects other services, you can run the **whitelist session-car bgp disable** command to disable whitelist session-CAR for BGP4+. In normal cases, you are advised to keep whitelist session-CAR for BGP4+ enabled.

### Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **whitelist session-car bgp { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } \***

Parameters of whitelist session-CAR for BGP4+ are configured.

In normal cases, you are advised to use the default values of these parameters.

### Step 3 Run commit

The configuration is committed.

----End

## Verifying the Configuration

Run the following command to check the previous configuration:

- Run the **display cpu-defend whitelist-v6 session-car bgpv6 statistics slot slot-id** command to check statistics about whitelist session-CAR for BGP4+ on a specified interface board.

To check the statistics within a specific period of time, run the **reset cpu-defend whitelist session-car bgpv6 statistics slot slot-id** command to clear the existing statistics about whitelist session-CAR for BGP4+ on the specified interface board; after a certain period of time, run the **display cpu-defend whitelist-v6 session-car bgpv6 statistics slot slot-id** command.

#### NOTE

The statistics about whitelist session-CAR for BGP4+ on a specified interface board cannot be restored after being cleared. Therefore, exercise caution when clearing the statistics.

## Configuring Whitelist Session-CAR for BMP

You can configure whitelist session-CAR for BMP to isolate bandwidth resources by session for BMP messages.

## Context

The function of whitelist session-CAR for BMP sets an independent CAR channel for each BMP session to ensure that the bandwidth of each BMP session is not preempted by other traffic (including traffic of other sessions of the same protocol and traffic of other protocols). When BMP messages form a traffic burst, you can adjust the bandwidth for each BMP session in whitelist session-CAR for BMP to ensure that BMP messages can be sent to the CPU properly.

#### NOTE

If the function becomes abnormal or affects other services, you can run the **whitelist session-car bmp disable** command to disable whitelist session-CAR for BMP. In normal cases, you are advised to keep whitelist session-CAR for BMP enabled.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **whitelist session-car bmp { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } \***

Parameters of whitelist session-CAR for BMP are configured.

In normal cases, you are advised to use the default values of these parameters.

### Step 3 Run commit

The configuration is committed.

----End

## Verifying the Configuration

Run the following command to check the previous configuration:

- Run the **display cpu-defend whitelist-v6 session-car bmpv6 statistics slot slot-id** command to check statistics about IPv6 whitelist session-CAR for BMP on a specified interface board.

To check the statistics within a specific period of time, run the **reset cpu-defend whitelist-v6 session-car bmpv6 statistics slot slot-id** command to clear the existing statistics about IPv6 whitelist session-CAR for BMP on the specified interface board; after a certain period of time, run the **display cpu-defend whitelist-v6 session-car bmpv6 statistics slot slot-id** command.



The statistics about whitelist session-CAR for BMP on a specified interface board cannot be restored after being cleared. Therefore, exercise caution before clearing them.

## Configuring Whitelist Session-CAR for RPKI

You can configure whitelist session-CAR for RPKI to isolate bandwidth resources by session for RPKI messages.

## Context

The function of whitelist session-CAR for RPKI sets an independent CAR channel for each RPKI session to ensure that the bandwidth of each RPKI session is not preempted by other traffic (including traffic of other sessions of the same protocol and traffic of other protocols). When RPKI messages form a traffic burst, you can adjust the bandwidth for each RPKI session in whitelist session-CAR for RPKI to ensure that RPKI messages can be sent to the CPU properly.



If the function becomes abnormal or affects other services, you can run the **whitelist session-car rpk disable** command to disable whitelist session-CAR for RPKI. In normal cases, you are advised to keep whitelist session-CAR for RPKI enabled.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **whitelist session-car rpk { cir cir-value | cbs cbs-value | pir pir-value | pbs pbs-value } \***

Parameters of whitelist session-CAR for RPKI are configured.

In normal cases, you are advised to use the default values of these parameters.

### Step 3 Run commit

The configuration is committed.

----End

## Verifying the Configuration

Run the following command to check the previous configuration:

- Run the **display cpu-defend whitelist-v6 session-car rpkiv6 statistics slot slot-id** command to check statistics about IPv6 whitelist session-CAR for RPKI on a specified interface board.

To check the statistics within a specific period of time, run the **reset cpu-defend whitelist-v6 session-car rpkiv6 statistics slot slot-id** command to clear the existing statistics about IPv6 whitelist session-CAR for RPKI on the specified interface board; after a certain period of time, run the **display cpu-defend whitelist-v6 session-car rpkiv6 statistics slot slot-id** command.

#### NOTE

The statistics about whitelist session-CAR for RPKI on a specified interface board cannot be restored after being cleared. Therefore, exercise caution before clearing them.

### 1.1.10.1.34 Configuration Examples for BGP4+

This chapter provides several BGP4+ configuration examples.

## Example for Configuring Basic BGP4+ Functions

Before building BGP4+ networks, you need to configure basic BGP4+ functions.

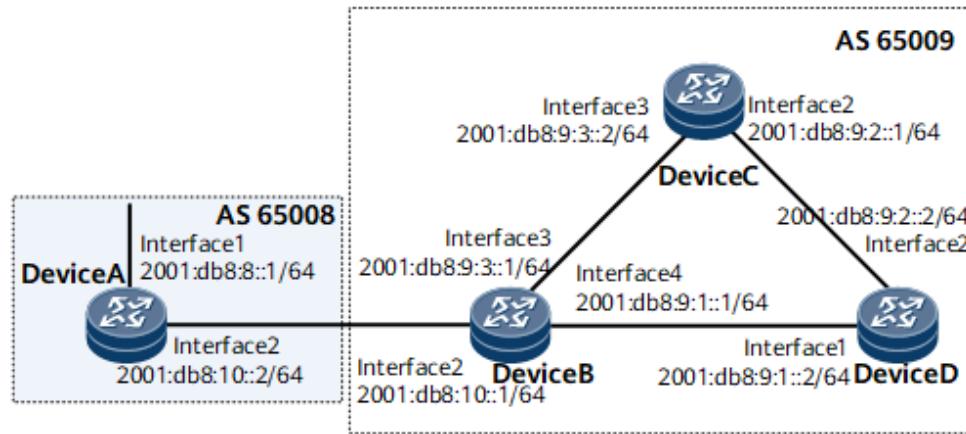
## Networking Requirements

In [Figure 1-459](#), there are two ASs: AS 65008 and AS 65009. Device A is in AS 65008, and Device B, Device C, and Device D are in AS 65009. BGP4+ must be configured to exchange routing information between the two ASs.

**Figure 1-459** Configuring basic BGP4+ functions

#### NOTE

In this example, interface1, interface2, interface3, and interface4 represent GE 1/0/0, GE 2/0/0, GE 3/0/0, and GE 4/0/0, respectively.



## Precautions

To improve security, you are advised to deploy BGP4+ security measures. For details, see "Configuring BGP4+ Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP4+ Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure IBGP connections between Device B, Device C, and Device D.
2. Configure an EBGP relationship between Device A and Device B.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs of Device A, Device B, Device C, and Device D
- AS numbers of Device A, Device B, Device C, and Device D

## Procedure

**Step 1** Configure an IPv6 address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure IBGP connections.

# Configure Device B.

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 2001:db8:9:1::2 as-number 65009
[*DeviceB-bgp] peer 2001:db8:9:3::2 as-number 65009
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:9:1::2 enable
[*DeviceB-bgp-af-ipv6] peer 2001:db8:9:3::2 enable
[*DeviceB-bgp-af-ipv6] network 2001:db8:9:1:: 64
[*DeviceB-bgp-af-ipv6] network 2001:db8:9:3:: 64
[*DeviceB-bgp-af-ipv6] commit
[~DeviceB-bgp-af-ipv6] quit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 65009
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 2001:db8:9:3::1 as-number 65009
[*DeviceC-bgp] peer 2001:db8:9:2::2 as-number 65009
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:9:3::1 enable
[*DeviceC-bgp-af-ipv6] peer 2001:db8:9:2::2 enable
[*DeviceC-bgp-af-ipv6] network 2001:db8:9:3:: 64
[*DeviceC-bgp-af-ipv6] network 2001:db8:9:2:: 64
[*DeviceC-bgp-af-ipv6] commit
[~DeviceC-bgp-af-ipv6] quit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 65009
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 2001:db8:9:1::1 as-number 65009
[*DeviceD-bgp] peer 2001:db8:9:2::1 as-number 65009
[*DeviceD-bgp] ipv6-family unicast
[*DeviceD-bgp-af-ipv6] peer 2001:db8:9:1::1 enable
[*DeviceD-bgp-af-ipv6] peer 2001:db8:9:2::1 enable
[*DeviceD-bgp-af-ipv6] network 2001:db8:9:2:: 64
[*DeviceD-bgp-af-ipv6] network 2001:db8:9:1:: 64
[*DeviceD-bgp-af-ipv6] commit
[~DeviceD-bgp-af-ipv6] quit
[~DeviceD-bgp] quit
```

### Step 3 Configure an EBGP connection.

# Configure Device A.

```
[~DeviceA] bgp 65008
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 2001:db8:10::1 as-number 65009
[*DeviceA-bgp] ipv6-family unicast
[*DeviceA-bgp-af-ipv6] peer 2001:db8:10::1 enable
[*DeviceA-bgp-af-ipv6] network 2001:db8:10:: 64
[*DeviceA-bgp-af-ipv6] network 2001:db8:8:: 64
[*DeviceA-bgp-af-ipv6] commit
[~DeviceA-bgp-af-ipv6] quit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 65009
[*DeviceB-bgp] peer 2001:db8:10::2 as-number 65008
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:10::2 enable
[*DeviceB-bgp-af-ipv6] network 2001:db8:10:: 64
[*DeviceB-bgp-af-ipv6] commit
[~DeviceB-bgp-af-ipv6] quit
[~DeviceB-bgp] quit
```

### Step 4 Verify the configuration.

# Check the status of BGP4+ connections.

```
[~DeviceB] display bgp ipv6 peer

BGP local router ID : 2.2.2.2
Local AS number : 65009
Total number of peers : 3 Peers in established state : 3

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
2001:db8:9:1::2 4 65009 8 9 0 00:05:37 Established 2
```

```
2001:db8:9:3::2 4 65009 2 2 0 00:00:09 Established 2
2001:db8:10::2 4 65008 9 7 0 00:05:38 Established 2
```

The preceding command output shows that BGP4+ connections have been established between Device B and other routers.

# Display the routing table of Device A.

```
[~DeviceA] display bgp ipv6 routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 6
*> Network : 2001:db8:8:: PrefixLen : 64
 NextHop : :: LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : i

*> Network : 2001:db8:9:1:: PrefixLen : 64
 NextHop : 2001:db8:10::1 LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : 65009 i

*> Network : 2001:db8:9:2:: PrefixLen : 64
 NextHop : 2001:db8:10::1 LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : 65009 i

*> Network : 2001:db8:9:3:: PrefixLen : 64
 NextHop : 2001:db8:10::1 LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : 65009 i

*> Network : 2001:db8:10:: PrefixLen : 64
 NextHop : :: LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : i

* NextHop : 2001:db8:10::1 LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : 65009 i
```

The preceding command output shows that Device A has learned routes from its peer in AS 65009. AS 65008 and AS 65009 can exchange routing information.

----End

## Configuration Files

- Device A configuration file

```

sysname DeviceA

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:8::1/64

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:10::2/64
```

```

bgp 65008
router-id 1.1.1.1
peer 2001:db8:10::1 as-number 65009

ipv4-family unicast
undo synchronization

ipv6-family unicast
network 2001:db8:8:: 64
network 2001:db8:10:: 64
peer 2001:db8:10::1 enable

return
```

- Device B configuration file

```

sysname DeviceB

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9:1::1/64

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:10::1/64

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9:3::1/64

bgp 65009
router-id 2.2.2.2
peer 2001:db8:9:1::2 as-number 65009
peer 2001:db8:9:3::2 as-number 65009
peer 2001:db8:10::2 as-number 65008

ipv4-family unicast
undo synchronization

ipv6-family unicast
network 2001:db8:9:1:: 64
network 2001:db8:9:3:: 64
network 2001:db8:10:: 64
peer 2001:db8:9:1::2 enable
peer 2001:db8:9:3::2 enable
peer 2001:db8:10::2 enable

return
```

- Device C configuration file

```

sysname DeviceC

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9:2::1/64

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9:3::2/64

bgp 65009
router-id 3.3.3.3
peer 2001:db8:9:2::2 as-number 65009
```

```
peer 2001:db8:9:3::1 as-number 65009
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
network 2001:db8:9:2:: 64
network 2001:db8:9:3:: 64
peer 2001:db8:9:2::2 enable
peer 2001:db8:9:3::1 enable
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9:1::2/64
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9:2::2/64
#
bgp 65009
router-id 4.4.4.4
peer 2001:db8:9:1::1 as-number 65009
peer 2001:db8:9:2::1 as-number 65009
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
network 2001:db8:9:1:: 64
network 2001:db8:9:2:: 64
peer 2001:db8:9:1::1 enable
peer 2001:db8:9:2::1 enable
#
return
```

## Example for Configuring BGP4+ Route Reflection

Configuring BGP4+ RRs simplifies the network configuration because IBGP peers do not need to be fully meshed.

## Networking Requirements

In [Figure 1-460](#), Device B receives an Update packet from its EBGP peer and forwards it to Device C. Device C is an RR and has two clients: Device B and Device D.

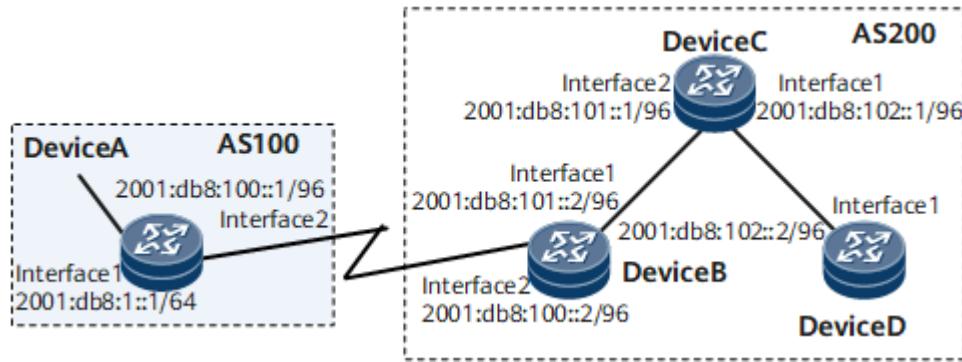
Device B and Device D do not need to establish an IBGP connection. After receiving a route update message from Device B, Device C reflects it to Device D. Similarly, after receiving a route update message from Device D, Device C reflects it to Device B.

**Figure 1-460** Network diagram of configuring BGP4+ route reflection



NOTE

In this example, interface1 and interface2 represent GE 1/0/0 and GE 2/0/0, respectively.



## Precautions

To improve security, you are advised to deploy BGP4+ security measures. For details, see "Configuring BGP4+ Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP4+ Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic BGP4+ functions on each router.
2. Configure an RR and its clients to establish IBGP connections.
3. Configure Device C as an RR, and then check its routing information.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs of Device A, Device B, Device C, and Device D
- AS numbers of Device A, Device B, Device C, and Device D

## Procedure

**Step 1** Configure an IPv6 address for each interface. For configuration details, see [Configuration Files](#) in this section.

**Step 2** Configure basic BGP4+ functions.

# Configure Device A.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 2001:db8:100::2 as-number 200
[*DeviceA-bgp] ipv6-family unicast
[*DeviceA-bgp-af-ipv6] peer 2001:db8:100::2 enable
[*DeviceA-bgp-af-ipv6] network 2001:db8:1:: 64
[*DeviceA-bgp-af-ipv6] commit
[~DeviceA-bgp-af-ipv6] quit
[~DeviceA-bgp] quit
```

# Configure Device B.

```
[~DeviceB] bgp 200
```

```
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 2001:db8:100::1 as-number 100
[*DeviceB-bgp] peer 2001:db8:101::1 as-number 200
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:100::1 enable
[*DeviceB-bgp-af-ipv6] peer 2001:db8:101::1 enable
[*DeviceB-bgp-af-ipv6] peer 2001:db8:101::1 next-hop-local
[*DeviceB-bgp-af-ipv6] commit
[~DeviceB-bgp-af-ipv6] quit
[~DeviceB-bgp] quit
```

# Configure Device C.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 2001:db8:101::2 as-number 200
[*DeviceC-bgp] peer 2001:db8:102::2 as-number 200
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:101::2 enable
[*DeviceC-bgp-af-ipv6] peer 2001:db8:102::2 enable
[*DeviceC-bgp-af-ipv6] network 2001:db8:101:: 96
[*DeviceC-bgp-af-ipv6] network 2001:db8:102:: 96
[*DeviceC-bgp-af-ipv6] commit
[~DeviceC-bgp-af-ipv6] quit
[~DeviceC-bgp] quit
```

# Configure Device D.

```
[~DeviceD] bgp 200
[*DeviceD-bgp] router-id 4.4.4.4
[*DeviceD-bgp] peer 2001:db8:102::1 as-number 200
[*DeviceD-bgp] ipv6-family unicast
[*DeviceD-bgp-af-ipv6] peer 2001:db8:102::1 enable
[*DeviceD-bgp-af-ipv6] commit
[~DeviceD-bgp-af-ipv6] quit
[~DeviceD-bgp] quit
```

### Step 3 Configure an RR.

# Configure Device C as an RR, with Device B and Device D as its clients.

```
[~DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:101::2 reflect-client
[*DeviceC-bgp-af-ipv6] peer 2001:db8:102::2 reflect-client
[*DeviceC-bgp-af-ipv6] commit
```

### Step 4 Verify the configuration.

# Check the routing table of Device B.

```
[~DeviceB] display bgp ipv6 routing-table

BGP Local router ID is 2.2.2.2
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 3
*> Network : 2001:db8:1:: PrefixLen : 64
 NextHop : 2001:db8:100::1 LocPrf :
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn: 100i
i Network : 2001:db8:101:: PrefixLen : 96
 NextHop : 2001:db8:101::1 LocPrf : 100
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn: i
*>i Network : 2001:db8:102:: PrefixLen : 96
```

```
NextHop : 2001:db8:101::1 LocPrf : 100
MED : 0 PrefVal : 0
Label :
Path/Ogn : i
```

# Display the routing table of Device D.

```
[~DeviceD] display bgp ipv6 routing-table
BGP Local router ID is 4.4.4.4
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 3
*>i Network : 2001:db8:1:: PrefixLen : 64
 NextHop : 2001:db8:101::2 LocPrf : 100
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : 100i
*>i Network : 2001:db8:101:: PrefixLen : 96
 NextHop : 2001:db8:102::1 LocPrf : 100
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : i
i Network : 2001:db8:102:: PrefixLen : 96
 NextHop : 2001:db8:102::1 LocPrf : 100
 MED : 0 PrefVal : 0
 Label :
 Path/Ogn : i
```

The preceding command output shows that Device D has learned from Device C the routes advertised by Device A.

----End

## Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:100::1/96
#
bgp 100
router-id 1.1.1.1
peer 2001:db8:100::2 as-number 200
#
ipv6-family unicast
undo synchronization
network 2001:db8:1:: 64
peer 2001:db8:100::2 enable
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet1/0/0
```

```
undo shutdown
ipv6 enable
ipv6 address 2001:db8:101::2/96
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:100::2/96
#
bgp 200
router-id 2.2.2.2
peer 2001:db8:100::1 as-number 100
peer 2001:db8:101::1 as-number 200
#
ipv6-family unicast
undo synchronization
peer 2001:db8:100::1 enable
peer 2001:db8:101::1 enable
peer 2001:db8:101::1 next-hop-local
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:102::1/96
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:101::1/96
#
bgp 200
router-id 3.3.3.3
peer 2001:db8:101::2 as-number 200
peer 2001:db8:102::2 as-number 200
#
ipv6-family unicast
undo synchronization
network 2001:db8:101:: 96
network 2001:db8:102:: 96
peer 2001:db8:101::2 enable
peer 2001:db8:101::2 reflect-client
peer 2001:db8:102::2 enable
peer 2001:db8:102::2 reflect-client
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:102::2/96
#
bgp 200
router-id 4.4.4.4
peer 2001:db8:102::1 as-number 200
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
```

```
peer 2001:db8:102::1 enable
#
return
```

## Example for Configuring BGP4+ Keychain

Configuring keychain authentication between BGP4+ peers can enhance the security of BGP4+ connections.

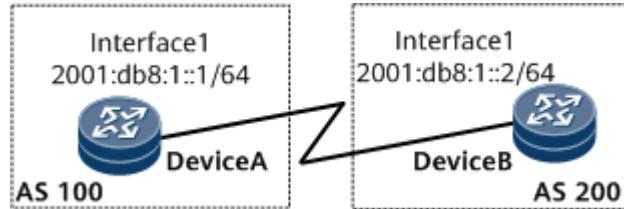
## Networking Requirements

On the network shown in [Figure 1-461](#), DeviceA belongs to AS 100, and DeviceB belongs to AS 200. BGP runs on the network, and BGP4+ keychain is used to protect EBGP connections against attacks.

**Figure 1-461** Networking diagram for configuring BGP4+ keychain



Interface1 in this example represents GE1/0/0.



## Precautions

During the configuration, note the following:

- Keychain authentication must be configured on both BGP4+ peers. The keychains configured at both ends must use the same encryption algorithm and password so that a TCP connection can be set up and BGP4+ messages can be exchanged properly.
- For security purposes, you are advised not to use weak security algorithms in this feature. If you need to use such an algorithm, run the **undo crypto weak-algorithm disable** command to enable the weak security algorithm function first.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Establish an EBGP connection between DeviceA and DeviceB.
2. Configure keychain authentication on DeviceA and DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of DeviceA and DeviceB
- Name of keychain authentication between DeviceA and DeviceB

- Passwords to be encrypted using the HMAC-SHA256 algorithm for the authentication on DeviceA and DeviceB

## Procedure

**Step 1** Configure an IP address for each interface. For configuration details, see Configuration Files in this section.

**Step 2** Configure EBGP connections.

# Configure DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 2001:db8:1::2 as-number 200
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 2001:db8:1::1 as-number 100
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

**Step 3** Configure a keychain.

# Configure DeviceA.

```
[~DeviceA] keychain AMode mode absolute
[*DeviceA-keychain] tcp-kind 179
[*DeviceA-keychain] tcp-algorithm-id hmac-sha-256 17
[*DeviceA-keychain] receive-tolerance 100
[*DeviceA-keychain] key-id 1
[*DeviceA-keychain-keyid-1] algorithm hmac-sha-256
[*DeviceA-keychain-keyid-1] key-string hello
[*DeviceA-keychain-keyid-1] send-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceA-keychain-keyid-1] receive-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceA-keychain-keyid-1] commit
[~DeviceA-keychain-keyid-1] quit
[~DeviceA-keychain] quit
```

# Configure DeviceB.

```
[~DeviceB] keychain AMode mode absolute
[*DeviceB-keychain] tcp-kind 179
[*DeviceB-keychain] tcp-algorithm-id hmac-sha-256 17
[*DeviceB-keychain] receive-tolerance 100
[*DeviceB-keychain] key-id 1
[*DeviceB-keychain-keyid-1] algorithm hmac-sha-256
[*DeviceB-keychain-keyid-1] key-string hello
[*DeviceB-keychain-keyid-1] send-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceB-keychain-keyid-1] receive-time 11:00 2009-12-24 to 12:00 2009-12-24
[*DeviceB-keychain-keyid-1] commit
[~DeviceB-keychain-keyid-1] quit
[~DeviceB-keychain] quit
```

**Step 4** Apply keychain authentication on the EBGP connection between DeviceA and DeviceB.

# Configure DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 2001:db8:1::2 keychain AMode
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] peer 2001:db8:1::1 keychain AMode
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

### Step 5 Verifying the Configuration

# On DeviceA, check the BGP connection status after keychain authentication is configured.

```
<~DeviceA> display bgp ipv6 peer
BGP local router ID : 1.1.1.1
Local AS number : 100
Total number of peers : 1 Peers in established state : 1

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
2001:db8:1::2 4 200 4 5 0 00:00:23 Established 1
```

You can view that the status of the BGP connection is Established after keychain authentication is configured.

# Run the **display keychain keychain-name** command on DeviceA to check the Key-id in **Active** state.

```
<~DeviceA> display keychain Amode
Keychain Information:

Keychain Name : amode
Timer Mode : Absolute
Receive Tolerance(min) : 100
Digest Length : 32
Time Zone : LMT
TCP Kind : 179
TCP Algorithm IDs :
 HMAC-MD5 : 5
 HMAC-SHA1-12 : 2
 HMAC-SHA1-20 : 6
 MD5 : 3
 SHA1 : 4
 HMAC-SHA-256 : 17
 SHA-256 : 8
 SM3 : 9
 AES-128-CMAC : 10
 HMAC-SHA-384 : 11
 HMAC-SHA-512 : 12
Number of Key IDs : 1
Active Send Key ID : 1
Active Receive Key IDs : 01
Default send Key ID : Not configured
```

Key ID Information:

```

Key ID : 1
Key string : *****
Algorithm : HMAC-SHA-256
SEND TIMER :
 Start time : 2013-08-10 10:00
 End time : 2022-10-28 12:00
 Status : Active
RECEIVE TIMER :
 Start time : 2013-08-10 10:00
 End time : 2022-10-28 12:00
 Status : Active
```

# Run the **display bgp ipv6 peer ipv6-address verbose** command to check whether the authentication type configured for the BGP peer is **Keychain(AMode)**.

```
<~DeviceA> display bgp ipv6 peer 2001:DB8:1::2 verbose
BGP Peer is 2001:DB8:1::2, remote AS 200
Type: EBGP link
BGP version 4, Remote router ID 2.2.2.2
Update-group ID: 3
BGP current state: Established, Up for 00h06m41s
BGP current event: KATimerExpired
BGP last state: OpenConfirm
BGP Peer Up count: 1
Received total routes: 1
Received active routes total: 0
Advertised total routes: 1
Port: Local - 179 Remote - 59233
Configured: Connect-retry Time: 32 sec
Configured: Min Hold Time: 0 sec
Configured: Active Hold Time: 180 sec Keepalive Time:60 sec
Received : Active Hold Time: 180 sec
Negotiated: Active Hold Time: 180 sec Keepalive Time:60 sec
Peer optional capabilities:
Peer supports bgp multi-protocol extension
Peer supports bgp route refresh capability
Peer supports bgp 4-byte-as capability
Address family IPv6 Unicast: advertised and received
Received: Total 11 messages
 Update messages 2
 Open messages 1
 KeepAlive messages 8
 Notification messages 0
 Refresh messages 0
Sent: Total 12 messages
 Update messages 2
 Open messages 2
 KeepAlive messages 8
 Notification messages 0
 Refresh messages 0
Authentication type configured: Keychain(amode)
Last keepalive received: 2013-08-15 17:42:11+00:00
Last keepalive sent : 2013-08-15 17:42:15+00:00
Last update received: 2013-08-15 17:36:08+00:00
Last update sent : 2013-08-15 17:36:08+00:00
No refresh received since peer has been configured
No refresh sent since peer has been configured
Minimum route advertisement interval is 30 seconds
Optional capabilities:
Route refresh capability has been enabled
4-byte-as capability has been enabled
Peer Preferred Value: 0
Memory Priority: medium
Routing policy configured:
No routing policy is configured
```

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

keychain AMMode mode absolute
receive-tolerance 100
tcp-kind 179
tcp-algorithm-id hmac-sha-256 1

key-id 1
algorithm hmac-sha-256
key-string cipher %##e^1}%%w;/C[M)OQc7"j+,2)}%#%#
send-time 11:00 2009-12-24 to 12:00 2009-12-24
```

```
receive-time 11:00 2009-12-24 to 12:00 2009-12-24
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1 64
#
bgp 100
router-id 1.1.1.1
peer 2001:db8:1::2 as-number 200
peer 2001:db8:1::2 keychain AMode
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
peer 2001:db8:1::2 enable
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
keychain AMode mode absolute
receive-tolerance 100
tcp-kind 179
tcp-algorithm-id hmac-sha-256 17
#
key-id 1
algorithm hmac-sha-256
key-string cipher %#%#ub(70WJ"^(i(kxPK@*fK,))}t%#%#
send-time 11:00 2009-12-24 to 12:00 2009-12-24
receive-time 11:00 2009-12-24 to 12:00 2009-12-24
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2 64
#
bgp 200
router-id 2.2.2.2
peer 2001:db8:1::1 as-number 100
peer 2001:db8:1::1 keychain AMode
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
peer 2001:db8:1::1 enable
#
return
```

## Example for Configuring BFD for BGP4+

If the primary link between two BGP4+ peers fails, BFD can quickly detect the failure and report it to BGP4+. This allows service traffic to be quickly switched to the backup link.

## Networking Requirements

In [Figure 1-462](#), DeviceA is in AS 100, and DeviceB and DeviceC are in AS 200. EBGP connections are established between DeviceA and DeviceB and between DeviceA and DeviceC.

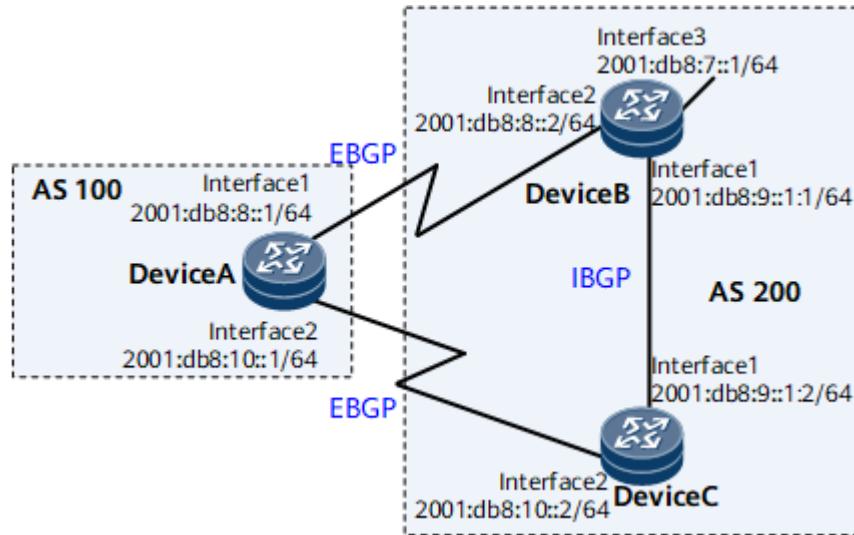
Service traffic is transmitted over the primary link DeviceA -> DeviceB. The link DeviceA -> DeviceC -> DeviceB serves as the backup link.

BFD is used to detect the BGP peer relationship between DeviceA and DeviceB. If the link between DeviceA and DeviceB fails, BFD can rapidly detect the failure and report it to BGP. This allows service traffic to be quickly switched to the backup link.

**Figure 1-462 Configuring BFD for BGP4+**

 **NOTE**

Interfaces 1 through 3 in this example are GE 1/0/0, GE 2/0/0, and GE 3/0/0, respectively.



## Precautions

To improve security, you are advised to deploy BGP4+ security measures. For details, see "Configuring BGP4+ Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP4+ Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic BGP4+ functions on each router.
2. Configure the MED attribute on DeviceB and DeviceC to control route selection, allowing traffic to be transmitted over the primary link between DeviceA and DeviceB.
3. Enable BFD on DeviceA and DeviceB.

## Data Preparation

To complete the configuration, you need the following data:

- Router IDs and AS numbers of DeviceA, DeviceB, and DeviceC

- IPv6 address of the remote end on a BFD session
- Minimum interval at which BFD Control packets are received and sent and the local detection multiplier

## Procedure

- Step 1** Configure an IPv6 address for each router. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure basic BGP4+ functions, establish EBGP connections between DeviceA and DeviceB and between DeviceA and DeviceC, and establish an IBGP connection between DeviceB and DeviceC.

# Configure DeviceA.

```
[~DeviceA] bgp 100
[*DeviceA-bgp] router-id 1.1.1.1
[*DeviceA-bgp] peer 2001:db8:8::2 as-number 200
[*DeviceA-bgp] peer 2001:db8:10::2 as-number 200
[*DeviceA-bgp] ipv6-family unicast
[*DeviceA-bgp-af-ipv6] peer 2001:db8:8::2 enable
[*DeviceA-bgp-af-ipv6] peer 2001:db8:10::2 enable
[*DeviceA-bgp-af-ipv6] commit
[~DeviceA-bgp-af-ipv6] quit
[~DeviceA-bgp] quit
```

# Configure DeviceB.

```
[~DeviceB] bgp 200
[*DeviceB-bgp] router-id 2.2.2.2
[*DeviceB-bgp] peer 2001:db8:8::1 as-number 100
[*DeviceB-bgp] peer 2001:db8:9::1:2 as-number 200
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:8::1 enable
[*DeviceB-bgp-af-ipv6] peer 2001:db8:9::1:2 enable
[*DeviceB-bgp-af-ipv6] network 2001:db8:7::1 64
[*DeviceB-bgp-af-ipv6] commit
[~DeviceB-bgp-af-ipv6] quit
[~DeviceB-bgp] quit
```

# Configure DeviceC.

```
[~DeviceC] bgp 200
[*DeviceC-bgp] router-id 3.3.3.3
[*DeviceC-bgp] peer 2001:db8:10::1 as-number 100
[*DeviceC-bgp] peer 2001:db8:9::1:1 as-number 200
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:10::1 enable
[*DeviceC-bgp-af-ipv6] peer 2001:db8:9::1:1 enable
[*DeviceC-bgp-af-ipv6] commit
[~DeviceC-bgp-af-ipv6] quit
[~DeviceC-bgp] quit
```

# Display information about the BGP peer relationship on DeviceA. The following command output shows that the BGP peer relationship has been established on the device.

```
<DeviceA> display bgp ipv6 peer

BGP local router ID : 1.1.1.1
Local AS number : 100
Total number of peers : 2 Peers in established state : 2

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
```

```
2001:db8:8::2 4 200 12 11 0 00:07:26 Established 0
2001:db8:10::2 4 200 12 12 0 00:07:21 Established 0
```

**Step 3** Configure BFD to detect the BGP peer relationship between DeviceA and DeviceB.

```
Enable BFD on DeviceA and establish a BFD session to detect the link between
DeviceA and DeviceB.
```

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] commit
```

```
Enable BFD on DeviceB and establish a BFD session to detect the link between
DeviceB and DeviceA.
```

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] commit
```

**Step 4** Configure the MED attribute.

Configure a route-policy to set the MEDs for the routes that DeviceB and DeviceC send to DeviceA.

```
Configure DeviceB.
```

```
[~DeviceB] route-policy 10 permit node 10
[*DeviceB-route-policy] apply cost 100
[*DeviceB-route-policy] quit
[*DeviceB] bgp 200
[*DeviceB-bgp] ipv6-family unicast
[*DeviceB-bgp-af-ipv6] peer 2001:db8:8::1 route-policy 10 export
[*DeviceB-bgp-af-ipv6] quit
[*DeviceB-bgp] quit
[*DeviceB] commit
```

```
Configure DeviceC.
```

```
[~DeviceC] route-policy 10 permit node 10
[*DeviceC-route-policy] apply cost 150
[*DeviceC-route-policy] quit
[*DeviceC] bgp 200
[*DeviceC-bgp] ipv6-family unicast
[*DeviceC-bgp-af-ipv6] peer 2001:db8:10::1 route-policy 10 export
[*DeviceC-bgp-af-ipv6] quit
[*DeviceC-bgp] quit
[*DeviceC] commit
```

```
Display all BGP routes on DeviceA.
```

```
<DeviceA> display bgp ipv6 routing-table

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 2
*> Network : 2001:db8:7:: PrefixLen : 64
 NextHop : 2001:db8:8::2 LocPrf :
 MED : 100 PrefVal : 0
 Label :
 Path/Ogn: 200 i
*
 NextHop : 2001:db8:10::2 LocPrf :
 MED : 150 PrefVal : 0
 Label :
 Path/Ogn: 200 i
```

The preceding command output shows that the next hop address of the BGP route to 2001:db8:7::1/64 is 2001:db8:8::2 and that traffic is transmitted on the primary link DeviceA → DeviceB.

- Step 5** Configure BFD, and set the interval at which BFD Control packets are received and sent and the local detection multiplier.

# Enable BFD on DeviceA, set the minimum interval at which BFD Control packets are received and sent to 100 ms, and set the local detection multiplier to 4.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bgp 100
[*DeviceA-bgp] peer 2001:db8:8::2 bfd enable
[*DeviceA-bgp] peer 2001:db8:8::2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceA-bgp] quit
[*DeviceA] commit
```

# Enable BFD on DeviceB, set the minimum interval at which BFD Control packets are received and sent to 100 ms, and set the local detection multiplier to 4.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] bgp 200
[*DeviceB-bgp] peer 2001:db8:8::1 bfd enable
[*DeviceB-bgp] peer 2001:db8:8::1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceB-bgp] commit
```

# Display all BFD sessions established by BGP on DeviceA.

```
<DeviceA> display bgp ipv6 bfd session all
```

```

Local_Address : 2001:db8:8::1
Peer_Address : 2001:db8:8::2
Tx-interval(ms): 100 Rx-interval(ms): 100
Multiplier : 4 Interface : GigabitEthernet1/0/0
Session-State : Up
Wtr-interval(m): 0

```

- Step 6** Verify the configuration.

# Run the **shutdown** command on GE 2/0/0 of DeviceB to simulate a fault in the primary link.

```
[~DeviceB] interface gigabitethernet 2/0/0
[~DeviceB-GigabitEthernet2/0/0] shutdown
```

# Check the BGP routing table on routerDeviceA.

```
<DeviceA> display bgp ipv6 routing-table
Total Number of Routes: 1

BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
 h - history, i - internal, s - suppressed, S - Stale
 Origin : i - IGP, e - EGP, ? - incomplete
 RPKI validation codes: V - valid, I - invalid, N - not-found

*> Network : 2001:db8:7:: PrefixLen : 64
 NextHop : 2001:db8:10::2 LocPrf :
 MED : 150 PrefVal : 0
 Label :
 Path/Ogn: 200 i
```

The preceding command output shows that the next hop address of the BGP route to 2001:db8:7::1/64 becomes 2001:db8:10::2 and that the backup link DeviceA → DeviceC → DeviceB takes effect after the primary link fails.

----End

## Configuration Files

- DeviceA configuration file

```

sysname DeviceA

bfd

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:8::1/64

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:10::1/64

bgp 100
router-id 1.1.1.1
peer 2001:db8:8::2 as-number 200
peer 2001:db8:8::2 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 2001:db8:8::2 bfd enable
peer 2001:db8:10::2 as-number 200

ipv4-family unicast
undo synchronization

ipv6-family unicast
undo synchronization
peer 2001:db8:8::2 enable
peer 2001:db8:10::2 enable

return
```

- DeviceB configuration file

```

sysname DeviceB

bfd

interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:8::2/64

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9::1:1/64

interface GigabitEthernet3/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:7::1/64

bgp 200
router-id 2.2.2.2
peer 2001:db8:8::1 as-number 100
peer 2001:db8:8::1 bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
peer 2001:db8:8::1 bfd enable
```

```
peer 2001:db8:9::1:2 as-number 200
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
network 2001:db8:7::1 64
peer 2001:db8:8::1 enable
peer 2001:db8:8::1 route-policy 10 export
peer 2001:db8:9::1:2 enable
#
route-policy 10 permit node 10
apply cost 100
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:9::1:2/64
#
interface GigabitEthernet2/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:10::2/64
#
bgp 200
router-id 3.3.3.3
peer 2001:db8:9::1:1 as-number 200
peer 2001:db8:10::1 as-number 100
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
peer 2001:db8:9::1:1 enable
peer 2001:db8:10::1 enable
peer 2001:db8:10::1 route-policy 10 export
#
route-policy 10 permit node 10
apply cost 150
#
return
```

## Example for Configuring BGP4+ 6PE

BGP4+ 6PE enables separated IPv6 networks to communicate with each other using the MPLS tunneling technology.

### Networking Requirements

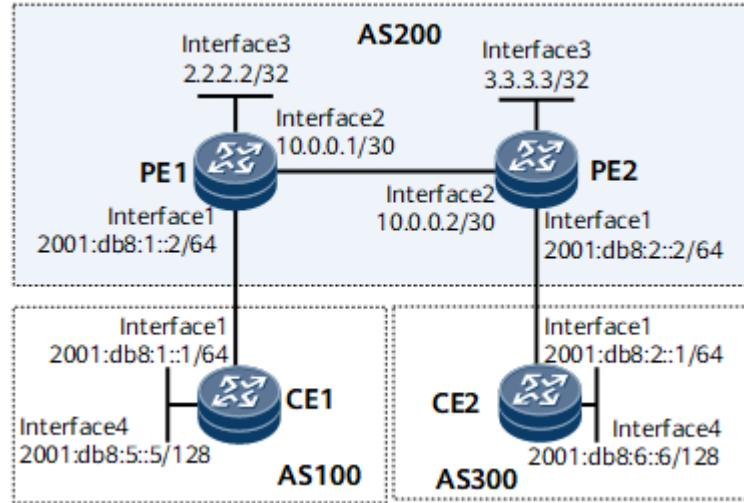
6PE enables IPv6 networks separated by IPv4/MPLS networks to communicate with each other.

In [Figure 1-463](#), the IPv6 network where CE1 resides and the IPv6 network where CE2 resides are connected by an IPv4/MPLS network in AS 200. A 6PE peer relationship must be established between PE1 and PE2 so that CE1 and CE2 can communicate with each other across the IPv4/MPLS network. The 6PE peers send IPv6 routes learned from their attached CEs to each other using MP-BGP, and forward IPv6 data over an LDP LSP.

**Figure 1-463 Configuring BGP4+ 6PE**

 **NOTE**

Interfaces 1 through 4 in this example represent GE 1/0/0, GE 2/0/0, Loopback 0, and Loopback 1, respectively.



## Precautions

To improve security, you are advised to deploy BGP4+ security measures. For details, see "Configuring BGP4+ Security." Keychain authentication is used as an example. For details, see "Example for Configuring BGP4+ Keychain."

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure OSPF on PE1 and PE2 to make them learn loopback interface addresses from each other.
2. Enable MPLS and MPLS LDP on the backbone network so that an LDP LSP can be established between PEs.
3. Establish a 6PE peer relationship between PE1 and PE2.
4. Configure loopback interface addresses on CE1 and CE2 to simulate directly connected subnets.
5. Configure BGP4+ on PEs and CEs to exchange IPv6 routes.

## Data Preparation

To complete the configuration, you need the following data:

- Router ID of each device
- Number of the AS where each device resides

## Procedure

- Step 1** Configure IPv4 and IPv6 addresses for each interface. For configuration details, see [Configuration Files](#) in this section.
- Step 2** Configure OSPF on PE1 and PE2 so that they learn loopback interface addresses from each other. For details, see [Configuration Files](#) in this section.
- Step 3** Enable MPLS and MPLS LDP on the backbone network so that an LDP LSP can be established between the PEs.

# Configure PE1.

```
[~PE1] mpls lsr-id 2.2.2.2
[*PE1] mpls
[*PE1-mpls] quit
[*PE1] mpls ldp
[*PE1-mpls-ldp] quit
[*PE1] interface gigabitethernet2/0/0
[*PE1-GigabitEthernet2/0/0] mpls
[*PE1-GigabitEthernet2/0/0] mpls ldp
[*PE1-GigabitEthernet2/0/0] quit
[*PE1] commit
```

# Configure PE2.

```
[~PE2] mpls lsr-id 3.3.3.3
[*PE2] mpls
[*PE2-mpls] quit
[*PE2] mpls ldp
[*PE2-mpls-ldp] quit
[*PE2] interface gigabitethernet2/0/0
[*PE2-GigabitEthernet2/0/0] mpls
[*PE2-GigabitEthernet2/0/0] mpls ldp
[*PE2-GigabitEthernet2/0/0] quit
[*PE2] commit
```

After completing the preceding configurations, run the **display mpls ldp session** command on each PE. The following command output shows that an LDP session has been established between the PEs.

```
[*PE1] display mpls ldp session
LDP Session(s) in Public Network
Codes: LAM(Label Advertisement Mode), SsnAge Unit(DDDD:HH:MM)
An asterisk (*) before a session means the session is being deleted.

PeerID Status LAM SsnRole SsnAge KASent/Rcv

3.3.3.3:0 Operational DU Passive 0000:00:35 143/199

TOTAL: 1 Session(s) Found.
```

- Step 4** Establish a 6PE peer relationship between the PEs.

# Configure PE1.

```
[~PE1] bgp 200
[*PE1-bgp] peer 3.3.3.3 as-number 200
[*PE1-bgp] peer 3.3.3.3 connect-interface LoopBack0
[*PE1-bgp] ipv6-family unicast
[*PE1-bgp-af-ipv6] import-route direct
[*PE1-bgp-af-ipv6] peer 3.3.3.3 enable
[*PE1-bgp-af-ipv6] peer 3.3.3.3 label-route-capability
[*PE1-bgp-af-ipv6] quit
[*PE1-bgp] quit
[*PE1] commit
```

# Configure PE2.

```
[~PE2] bgp 200
[*PE2-bgp] peer 2.2.2.2 as-number 200
[*PE2-bgp] peer 2.2.2.2 connect-interface LoopBack0
[*PE2-bgp] ipv6-family unicast
[*PE2-bgp-af-ipv6] import-route direct
[*PE2-bgp-af-ipv6] peer 2.2.2.2 enable
[*PE2-bgp-af-ipv6] peer 2.2.2.2 label-route-capability
[*PE2-bgp-af-ipv6] commit
[~PE2-bgp-af-ipv6] quit
[~PE2-bgp] quit
[~PE2] commit
```

# After completing the preceding configurations, run the **display bgp ipv6 peer** command on each PE. The following command output shows that a BGP peer relationship has been established between the PEs.

```
[~PE1] display bgp ipv6 peer
```

```
BGP local router ID : 10.0.0.1
Local AS number : 200
Total number of peers : 2 Peers in established state : 2

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
3.3.3.3 4 200 1248 1342 0 18:06:28 Established 1
```

**Step 5** Configure BGP4+ on PEs and CEs to exchange IPv6 routes.

# Configure CE1.

```
[~CE1] bgp 100
[*CE1-bgp] router-id 5.5.5.5
[*CE1-bgp] peer 2001:db8:1::2 as-number 200
[*CE1-bgp] ipv6-family unicast
[*CE1-bgp-af-ipv6] peer 2001:db8:1::2 enable
[*CE1-bgp-af-ipv6] network 2001:db8:5::5 128
[*CE1-bgp-af-ipv6] commit
[~CE1-bgp-af-ipv6] quit
[~CE1-bgp] quit
```

# Configure PE1.

```
[~PE1] bgp 200
[*PE1-bgp] peer 2001:db8:1::1 as-number 100
[*PE1-bgp] ipv6-family unicast
[*PE1-bgp-af-ipv6] peer 2001:db8:1::1 enable
[*PE1-bgp-af-ipv6] commit
[~PE1-bgp-af-ipv6] quit
[~PE1-bgp] quit
```

# Configure PE2.

```
[~PE2] bgp 200
[*PE2-bgp] peer 2001:db8:2::1 as-number 300
[*PE2-bgp] ipv6-family unicast
[*PE2-bgp-af-ipv6] peer 2001:db8:2::1 enable
[*PE2-bgp-af-ipv6] commit
[~PE2-bgp-af-ipv6] quit
[~PE2-bgp] quit
```

# Configure CE2.

```
[~CE2] bgp 300
[*CE2-bgp] router-id 6.6.6.6
[*CE2-bgp] peer 2001:db8:2::2 as-number 200
[*CE2-bgp] ipv6-family unicast
[*CE2-bgp-af-ipv6] peer 2001:db8:2::2 enable
```

```
[*CE2-bgp-af-ipv6] network 2001:db8:6::128
[*CE2-bgp-af-ipv6] commit
[*CE2-bgp-af-ipv6] quit
[~CE2-bgp] quit
```

After completing the preceding configurations, run the **display bgp ipv6 peer** command on each PE or CE. The following command output shows that the BGP peer relationship has been established between each PE and the corresponding CE.

In the following example, the command output on PE1 is used.

```
[~PE1] display bgp ipv6 peer

BGP local router ID : 10.0.0.1
Local AS number : 100
Total number of peers : 2 Peers in established state : 2

Peer V AS MsgRcvd MsgSent OutQ Up/Down State PrefRcv
3.3.3.3 4 200 59 60 0 00:35:46 Established 1
2001:db8:1::1 4 100 40 45 0 00:06:16 Established 1
```

### Step 6 Verify the configuration.

After the preceding configurations are complete, CEs can learn the routes to each other's loopback interface, and ping each other.

In the following example, the command output on CE1 is used.

```
[~CE1] display ipv6 routing-table

Routing Table : _public_
Destinations : 8 Routes : 8

Destination : ::1 PrefixLength : 128
NextHop : ::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : InLoopBack0 Flags : D

Destination : ::FFFF:127.0.0.0 PrefixLength : 104
NextHop : ::FFFF:127.0.0.0 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : InLoopBack0 Flags : D

Destination : ::FFFF:127.0.0.0 PrefixLength : 128
NextHop : ::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : InLoopBack0 Flags : D

Destination : 2001:db8:1:: PrefixLength : 64
NextHop : 2001:db8:1::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : D

Destination : 2001:db8:1::1 PrefixLength : 128
NextHop : ::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : D

Destination : 2001:db8:5::5 PrefixLength : 128
NextHop : ::1 Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : LoopBack2 Flags : D

Destination : 2001:db8:6::6 PrefixLength : 128
```

```
NextHop : 2001:db8:1::2 Preference : 255
Cost : 0 Protocol : BGP
RelayNextHop : 2001:db8:1::2 TunnelID : 0x0
Interface : GigabitEthernet1/0/0 Flags : RD

Destination : FE80:: PrefixLength : 10
NextHop : :: Preference : 0
Cost : 0 Protocol : Direct
RelayNextHop : :: TunnelID : 0x0
Interface : NULL0 Flags : D
<CE1> ping ipv6 -a 2001:db8:5::5 2001:db8:6::6

PING 2001:db8:6::6 : 56 data bytes, press CTRL_C to break
Reply from 2001:db8:6::6
bytes=56 Sequence=1 hop limit=62 time=8 ms
Reply from 2001:db8:6::6
bytes=56 Sequence=2 hop limit=62 time=2 ms
Reply from 2001:db8:6::6
bytes=56 Sequence=3 hop limit=62 time=4 ms
Reply from 2001:db8:6::6
bytes=56 Sequence=4 hop limit=62 time=3 ms
Reply from 2001:db8:6::6
bytes=56 Sequence=5 hop limit=62 time=4 ms
---2001:db8:6::6 ping statistics---
5 packet(s) transmitted
5 packet(s) received
0.00% packet loss
round-trip min/avg/max=2/4/8 ms
```

After 6PE is configured, the IPv6 network where CE1 resides and the IPv6 network where CE2 resides can communicate through the IPv4/MPLS network.

----End

## Configuration Files

- CE1 configuration file

```

sysname CE1
#
interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::1/64
#
interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:5::5/128
#
bgp 100
router-id 5.5.5.5
peer 2001:db8:1::2 as-number 200
#
ipv4-family unicast
undo synchronization
#
ipv6-family unicast
undo synchronization
network 2001:db8:5::5 128
peer 2001:db8:1::2 enable
#
return
```

- PE1 configuration file

```

sysname PE1
#
mpls lsr-id 2.2.2.2
```

```

mpls

mpls ldp

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:1::2/64

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.0.1 255.255.255.252
mpls
mpls ldp

interface LoopBack0
ip address 2.2.2.2 255.255.255.255

bgp 200
peer 3.3.3.3 as-number 200
peer 3.3.3.3 connect-interface LoopBack0
peer 2001:db8:1::1 as-number 100

ipv4-family unicast
undo synchronization
peer 3.3.3.3 enable

ipv6-family unicast
undo synchronization
import-route direct
peer 3.3.3.3 enable
peer 3.3.3.3 label-route-capability
peer 2001:db8:1::1 enable

ospf 1
area 0.0.0.0
network 2.2.2.2 0.0.0.0
network 10.0.0.0 0.0.0.3

return
```

- PE2 configuration file

```

sysname PE2

mpls lsr-id 3.3.3.3

mpls

mpls ldp

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::2/64

interface GigabitEthernet2/0/0
undo shutdown
ip address 10.0.0.2 255.255.255.252
mpls
mpls ldp

interface LoopBack0
ip address 3.3.3.3 255.255.255.255

bgp 200
peer 2.2.2.2 as-number 200
peer 2.2.2.2 connect-interface LoopBack0
peer 2001:db8:2::1 as-number 300
```

```

ipv4-family unicast
undo synchronization
peer 2.2.2.2 enable

ipv6-family unicast
undo synchronization
import-route direct
peer 2.2.2.2 enable
peer 2.2.2.2 label-route-capability
peer 2001:db8:2::1 enable

ospf 1
area 0.0.0.0
network 3.3.3.3 0.0.0.0
network 10.0.0.0 0.0.0.3

return
```

- CE2 configuration file

```

sysname CE2

interface GigabitEthernet1/0/0
undo shutdown
ipv6 enable
ipv6 address 2001:db8:2::1/64

interface LoopBack1
ipv6 enable
ipv6 address 2001:db8:6::6/128

bgp 300
router-id 6.6.6.6
peer 2001:db8:2::2 as-number 200

ipv4-family unicast
undo synchronization

ipv6-family unicast
undo synchronization
network 2001:db8:6::6 128
peer 2001:db8:2::2 enable

return
```

## 1.1.11 Routing Policy Configuration

### 1.1.11.1 Routing Policy Description

#### 1.1.11.1.1 Overview of Routing Policy

##### Definition

Routing policies are used to filter routes and control how routes are received and advertised. If route attributes, such as reachability, are changed, the path along which network traffic passes changes accordingly.

## Purpose

When advertising, receiving, and importing routes, the router implements certain routing policies based on actual networking requirements to filter routes and change the route attributes. Routing policies serve the following purposes:

- Control route advertising  
Only routes that match the rules specified in a policy are advertised.
- Control route receiving  
Only the required and valid routes are received, which reduces the routing table size and improves network security.
- Filter and control imported routes  
A routing protocol may import routes discovered by other routing protocols. Only routes that satisfy certain conditions are imported to meet the requirements of the protocol.
- Modify attributes of specified routes  
To enrich routing information, a routing protocol may import routing information discovered by other routing protocols. Only the routing information that satisfies the conditions is imported. Some attributes of the imported routing information are changed to meet the requirements of the routing protocol.

## Benefits

Routing policies have the following benefits:

- Control the routing table size, saving system resources.
- Control route receiving and advertising, improving network security.
- Modify attributes of routes for proper traffic planning, improving network performance.

## Differences Between the Routing Policy and Policy-based Routing

Unlike the routing mechanism that searches the forwarding table for matching routes based on the destination addresses of IP packets, policy-based routing (PBR) is based on the user-defined routing policies. PBR selects routes based on the user-defined routing policies, with reference to the source IP addresses and lengths of incoming packets. PBR can be used to improve security and implement load balancing.

A routing policy and PBR have different mechanisms. **Table 1-160** shows the differences between them.

**Table 1-160** Differences between the routing policy and PBR

| Routing Policy                                                        | Policy-based Routing                                                                                                                                           |
|-----------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Forwards packets based on destination addresses in the routing table. | Forwards packets based on the policy. The device searches the routing table for packet forwarding only after packets fail to be forwarded based on the policy. |

| Routing Policy                                                           | Policy-based Routing                                                                          |
|--------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| Based on the control plane, serves routing protocols and routing tables. | Based on the forwarding plane, serves forwarding.                                             |
| Combines with a routing protocol to form a policy.                       | Needs to be configured hop by hop to ensure that packets are forwarded based on the policies. |
| Is configured using the <b>route-policy</b> command.                     | Is configured using the <b>policy-based-route</b> command.                                    |

### 1.1.11.1.2 Understanding Routing Policies

#### Overview of Routing Policies

Routing policies are implemented in the following steps:

1. Define rules: The rules that contain characteristics of routes to which routing policies are applied need to be defined. Specifically, you need to define a set of matching rules regarding different attributes of routing information, such as the destination address and the source IP address of the device that advertises routes. A filter, the core of a routing policy, is used to define a set of matching rules.
2. Apply rules: The rules are used in a routing policy to advertise, accept, and import routes.

#### Filter

By using filters, you can define matching rules for a group of routing policies. The NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X provides multiple types of filters for routing policies. **Table 1-161** lists the filters supported by the device and their application scopes and matching conditions.

**Table 1-161** Filters supported by the device

| Filter                                    | Application Scope         | Matching Rules                                                                                            |
|-------------------------------------------|---------------------------|-----------------------------------------------------------------------------------------------------------|
| <a href="#">Access control list (ACL)</a> | Dynamic routing protocols | Inbound interface, source or destination IP address, protocol type, and source or destination port number |
| <a href="#">IP prefix list</a>            | Dynamic routing protocols | Source and destination IP addresses and next hop address                                                  |
| <a href="#">AS_Path</a>                   | BGP                       | AS_Path attribute                                                                                         |
| <a href="#">Community</a>                 | BGP                       | Community attribute                                                                                       |

| Filter                   | Application Scope         | Matching Rules                                                                                                                                                                                                                               |
|--------------------------|---------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Large-community          | BGP                       | Large-community attribute                                                                                                                                                                                                                    |
| Extended community       | VPN                       | Extended community attribute                                                                                                                                                                                                                 |
| Route distinguisher (RD) | VPN                       | RD attribute                                                                                                                                                                                                                                 |
| Route-Policy             | Dynamic routing protocols | Destination address, next hop address, cost, interface information, route type, ACL, IP prefix list, AS_Path filter, community filter, extended community filter, L2VNI list, L3VNI list, MAC address list, Ethernet tag list, and RD filter |

The ACL, IP prefix list, AS\_Path filter, Large-community filter, community filter, extended community filter, and RD filter can only be used to filter routes and cannot be used to modify the attributes of matched routes. A route-policy is a comprehensive filter, and it can use the matching rules of the ACL, IP prefix list, AS\_Path filter, community filter, extended community filter, and RD filter to filter routes. In addition, attributes of the matched routes can be modified. The following sections describe the filters in more detail.

## ACL

An ACL is a set of sequential filtering rules. Users can define rules based on packet information, such as inbound interfaces, source or destination IP addresses, protocol types, or source or destination port numbers and specify an action to deny or permit packets. After an ACL is configured, the system classifies received packets based on the rules defined in the ACL and denies or permits the packets accordingly.

An ACL only classifies packets based on defined rules and filters packets only after it is applied to a routing policy.

ACLs are classified as the ACLs that apply to IPv4 routes or those that apply to IPv6 routes. Based on the usage, ACLs are classified as interface-based ACLs, basic ACLs, or advanced ACLs. Users can specify the IP address and subnet address range in an ACL to match the source IP address, destination network segment address, or the next hop address of a route.

ACLs can be configured on network devices, such as access and core devices, to improve network security and stability. For example:

- Protect the devices against IP, TCP, and Internet Control Message Protocol (ICMP) packet attacks.
- Control network access. For example, ACLs can be used to control the access of enterprise network users to external networks, the specific network resources that users can access, and the period for which users can access networks.

- Limit network traffic and improve network performance. For example, ACLs can be used to limit bandwidth for upstream and downstream traffic, charge for the bandwidth that users have applied for, and fully use high-bandwidth network resources.

For details about ACL features, see [ACL Description](#).

## IP Prefix List

An IP prefix list contains a group of route filtering rules. Users can specify the prefix and mask length range to match the destination network segment address or the next hop address of a route. An IP prefix list is used to filter routes that are advertised and received by various dynamic routing protocols.

An IP prefix list is easier and more flexible than an ACL. However, if a large number of routes with different prefixes need to be filtered, configuring an IP prefix list to filter the routes is complex.

IP prefix lists are classified as IPv4 prefix lists that apply to IPv4 routes or IPv6 prefix lists that apply to IPv6 routes. IPv4 prefix lists and IPv6 prefix lists share the same implementation.. An IP prefix list filters routes based on the mask length or mask length range.

- Mask length: An IP prefix list filters routes based on IP address prefixes. An IP address prefix is defined by an IP address and the mask length. For example, for route 10.1.1.1/16, the mask length is 16 bits, and the valid prefix is 16 bits (10.1.0.0).
- Mask length range: Routes with the IP address prefix and mask length within the range defined in the IP prefix list meet the matching rules.

### NOTE

0.0.0.0 is a wildcard address. If the IP prefix is 0.0.0.0, specify either a mask or a mask length range, with the following results:

- If a mask is specified, all routes with the mask are permitted or denied.
- If a mask length range is specified, all routes with the mask length in the range are permitted or denied.

The following table describes the implementation of route matching rules when the preceding wildcard address is used.

**Table 1-162** Implementation of wildcard address-based route matching rules (IPv4)

| Whether <i>greater-equal</i> and <i>less-equal</i> Are Configured | Condition                                                                                                     | Matching Result                                                                               | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Neither <i>greater-equal</i> nor <i>less-equal</i> exists.        | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and 0, respectively.               | Matches only the default IPv4 route.                                                          | An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br><code>ip ip-prefix aa index 10 permit 1.1.1.1 0</code><br><b>Error:</b> Failed to add the address prefix list 0.0.0.0/0, because the destination address and mask do not match.<br><br>Correct configuration:<br><code>ip ip-prefix aa index 10 permit 0.0.0.0 0</code><br><br>Matching result: Only the default route is permitted.                                                                                          |
|                                                                   | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and X (non-0 value), respectively. | Matches all routes with the mask length of X.                                                 | Pre-processing:<br><code>ip ip-prefix aa index 10 permit 0.0.1.1 16</code><br><br>Post-processing:<br><code>ip ip-prefix aa index 10 permit 0.0.0.0 16</code><br><br>Matching result: The routes with the mask length of 16 are permitted.                                                                                                                                                                                                                                                                                |
| <i>greater-equal</i> exists, but <i>less-equal</i> does not.      | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and 0, respectively.               | Matches all the routes whose mask length is within the range from <i>greater-equal</i> to 32. | An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br><code>ip ip-prefix aa index 10 permit 1.1.1.1 0 greater-equal 16</code><br><b>Error:</b> Failed to add the address prefix list 0.0.0.0/0, because the destination address and mask do not match.<br><br>Correct configuration:<br><code>ip ip-prefix aa index 10 permit 0.0.0.0 0 greater-equal 16 less-equal 32</code><br><br>Matching result: The routes whose mask length is within the range from 16 to 32 are permitted. |

| Whether<br><i>greater-equal</i><br>and<br><i>less-equal</i><br>Are<br>Configured | Condition                                                                                                     | Matching<br>Result                                                                            | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                  | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and X (non-0 value), respectively. | Matches all the routes whose mask length is within the range from <i>greater-equal</i> to 32. | <p>Pre-processing:<br/>ip ip-prefix aa index 10 permit 0.0.1.1 16 greater-equal 20</p> <p>Post-processing:<br/>ip ip-prefix aa index 10 permit 0.0.0.0 16 greater-equal 20 less-equal 32</p> <p>Matching result: The routes whose mask length is within the range from 20 to 32 are permitted.</p>                                                                                                                                                                                             |
| <i>greater-equal</i> does not exist, but <i>less-equal</i> does.                 | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and 0, respectively.               | Matches all the routes whose mask length is within the range from 0 to <i>less-equal</i> .    | <p>An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br/>ip ip-prefix aa index 10 permit 1.1.1.1 0 less-equal 30</p> <p><b>Error:</b> Failed to add the address prefix list 0.0.0.0/0, because the destination address and mask do not match.</p> <p><b>Correct configuration:</b><br/>ip ip-prefix aa index 10 permit 0.0.0.0 0 less-equal 30</p> <p>Matching result: The routes whose mask length is within the range from 0 to 30 are permitted.</p> |
|                                                                                  | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and X (non-0 value), respectively. | Matches all the routes whose mask length is within the range from X to <i>less-equal</i> .    | <p>Pre-processing:<br/>ip ip-prefix aa index 10 permit 0.0.1.1 16 less-equal 30</p> <p>Post-processing:<br/>ip ip-prefix aa index 10 permit 0.0.0.0 16 greater-equal 16 less-equal 30</p> <p>Matching result: The routes whose mask length is within the range from 16 to 30 are permitted.</p>                                                                                                                                                                                                |

| Whether <i>greater-equal</i> and <i>less-equal</i> Are Configured | Condition                                                                                                     | Matching Result                                                                                               | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Both <i>greater-equal</i> and <i>less-equal</i> exist.            | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and 0, respectively.               | Matches all the routes whose mask length is within the range from <i>greater-equal</i> to <i>less-equal</i> . | <p>An IP prefix list cannot be configured if the prefix and mask do not match. For example:</p> <pre>ip ip-prefix aa index 10 permit 1.1.1.1 0 greater-equal 5 less-equal 30 Error: Failed to add the address prefix list 0.0.0.0/0, because the destination address and mask do not match.</pre> <p>Correct configuration:</p> <pre>ip ip-prefix aa index 10 permit 0.0.0.0 0 greater-equal 5 less-equal 30</pre> <p>Matching result: The routes whose mask length is within the range from 5 to 30 are permitted.</p> |
|                                                                   | The post-processing <i>ipv4-address</i> and <i>mask-length</i> are 0.0.0.0 and X (non-0 value), respectively. | Matches all the routes whose mask length is within the range from <i>greater-equal</i> to <i>less-equal</i> . | <p>Pre-processing:</p> <pre>ip ip-prefix aa index 10 permit 0.0.1.1 16 greater-equal 20 less-equal 30</pre> <p>Post-processing:</p> <pre>ip ip-prefix aa index 10 permit 0.0.0.0 16 greater-equal 20 less-equal 30</pre> <p>Matching result: The routes whose mask length is within the range from 20 to 30 are permitted.</p>                                                                                                                                                                                          |

**Table 1-163** Implementation of wildcard address-based route matching rules (IPv6)

| Whether <i>greater-equal</i> and <i>less-equal</i> Are Configured | Condition                                                                                                  | Matching Result                                                                                       | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Neither <i>greater-equal</i> nor <i>less-equal</i> exists.        | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and 0, respectively.               | Matches only the default IPv6 route.                                                                  | An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br><code>ip ipv6-prefix aa index 10 permit 1::1 0</code><br><b>Error:</b> Failed to add the address prefix list ::/0, because the destination address and mask do not match.<br><br>Correct configuration:<br><code>ip ipv6-prefix aa index 10 permit :: 0</code><br><br>Matching result: Only the default IPv6 route is permitted.                                                                                              |
|                                                                   | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and X (non-0 value), respectively. | Matches all IPv6 routes with the prefix length of X.                                                  | Pre-processing:<br><code>ip ipv6-prefix aa index 10 permit ::1:1 96</code><br><br>Post-processing:<br><code>ip ipv6-prefix aa index 10 permit :: 96</code><br><br>Matching result: The IPv6 routes with the prefix length of 96 are permitted.                                                                                                                                                                                                                                                                            |
| <i>greater-equal</i> exists, but <i>less-equal</i> does not.      | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and 0, respectively.               | Matches all the IPv6 routes whose prefix length is within the range from <i>greater-equal</i> to 128. | An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br><code>ip ipv6-prefix aa index 10 permit 1::1 0 greater-equal 16</code><br><b>Error:</b> Failed to add the address prefix list ::/0, because the destination address and mask do not match.<br><br>Correct configuration:<br><code>ip ipv6-prefix aa index 10 permit :: 0 greater-equal 16 less-equal 128</code><br><br>Matching result: The IPv6 routes whose prefix length is within the range from 16 to 128 are permitted. |

| Whether<br><i>greater-equal</i><br>and<br><i>less-equal</i><br>Are<br>Configured | Condition                                                                                                  | Matching<br>Result                                                                                    | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                  | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and X (non-0 value), respectively. | Matches all the IPv6 routes whose prefix length is within the range from <i>greater-equal</i> to 128. | <p>Pre-processing:<br/>ip ipv6-prefix aa index 10 permit ::1:1 96 greater-equal 120</p> <p>Post-processing:<br/>ip ipv6-prefix aa index 10 permit :: 96 greater-equal 120 less-equal 128</p> <p>Matching result: The IPv6 routes whose prefix length is within the range from 120 to 128 are permitted.</p>                                                                                                                                                                                     |
| <i>greater-equal</i> does not exist, but <i>less-equal</i> does.                 | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and 0, respectively.               | Matches all the IPv6 routes whose prefix length is within the range from 0 to <i>less-equal</i> .     | <p>An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br/>ip ipv6-prefix aa index 10 permit 1::1 0 less-equal 120</p> <p><b>Error:</b> Failed to add the address prefix list ::/0, because the destination address and mask do not match.</p> <p><b>Correct configuration:</b><br/>ip ipv6-prefix aa index 10 permit :: 0 less-equal 120</p> <p>Matching result: The IPv6 routes whose prefix length is within the range from 0 to 120 are permitted.</p> |
|                                                                                  | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and X (non-0 value), respectively. | Matches all the IPv6 routes whose prefix length is within the range from X to <i>less-equal</i> .     | <p>Pre-processing:<br/>ip ipv6-prefix aa index 10 permit ::1:1 96 less-equal 120</p> <p>Post-processing:<br/>ip ipv6-prefix aa index 10 permit :: 96 greater-equal 96 less-equal 120</p> <p>Matching result: The IPv6 routes whose prefix length is within the range from 96 to 120 are permitted.</p>                                                                                                                                                                                          |

| Whether <i>greater-equal</i> and <i>less-equal</i> Are Configured | Condition                                                                                                  | Matching Result                                                                                                      | Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Both <i>greater-equal</i> and <i>less-equal</i> exist.            | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and 0, respectively.               | Matches all the IPv6 routes whose prefix length is within the range from <i>greater-equal</i> to <i>less-equal</i> . | An IP prefix list cannot be configured if the prefix and mask do not match. For example:<br><code>ip ipv6-prefix aa index 10 permit 1::1 0 greater-equal 5 less-equal 30</code><br><b>Error:</b> Failed to add the address prefix list ::/0, because the destination address and mask do not match.<br><br>Correct configuration:<br><code>ip ipv6-prefix aa index 10 permit :: 0 greater-equal 5 less-equal 30</code><br><br>Matching result: The IPv6 routes whose prefix length is within the range from 5 to 30 are permitted. |
|                                                                   | The post-processing <i>ipv6-address</i> and <i>prefix-length</i> are :: and X (non-0 value), respectively. | Matches all the IPv6 routes whose prefix length is within the range from <i>greater-equal</i> to <i>less-equal</i> . | Pre-processing:<br><code>ip ipv6-prefix aa index 10 permit ::1:1 96 greater-equal 120 less-equal 124</code><br><br>Post-processing:<br><code>ip ipv6-prefix aa index 10 permit :: 96 greater-equal 120 less-equal 124</code><br><br>Matching result: The IPv6 routes whose prefix length is within the range from 120 to 124 are permitted.                                                                                                                                                                                        |

## AS\_Path

An AS\_Path filter is used to filter BGP routes based on AS\_Path attributes contained in BGP routes. The AS\_Path attribute is used to record in distance-vector (DV) order the numbers of all ASs through which a BGP route passes from the local end to the destination. Therefore, AS\_Path attributes can be used to filter BGP routes.

The matching condition of an AS\_Path is specified using a regular expression. For example, ^30 indicates that only the AS\_Path attribute starting with 30 is matched. Using a regular expression can simplify configurations. For details about regular expressions, see Configuration Guide - Basic Configurations.

 NOTE

The AS\_Path attribute is a private attribute of BGP and is therefore used to filter BGP routes only. For details about the AS\_Path attribute, see [BGP Fundamentals](#).

Regular expression matching is intensive processing of CPU computing. When a large number of regular expressions are configured in a policy to match a route attribute and the length of the route attribute is long, the processing performance of the policy deteriorates. To improve the processing performance of the route-policy, decrease the number of regular expressions or use non-regular expression matching commands.

## Community

A community filter is used to filter BGP routes based on the community attributes contained in BGP routes. The community attribute is a group of destination addresses with the same characteristics. Therefore, community attributes can be used to filter BGP routes.

In addition to the well-known community attributes, users can define community attributes using digits. The matching condition of a community filter can be specified using a community ID or a regular expression.

 NOTE

Like AS\_Path filters, community filters are used to filter only BGP routes because the community attribute is also a private attribute of BGP. For details about the community attribute, see [Community Attribute](#).

Regular expression matching is intensive processing of CPU computing. When a large number of regular expressions are configured in a policy to match a route attribute and the length of the route attribute is long, the processing performance of the policy deteriorates. To improve the processing performance of the route-policy, decrease the number of regular expressions or use non-regular expression matching commands.

## Large-community

A large-community filter is used to filter BGP routes based on large-community attributes contained in BGP routes. The large-community attribute is an extended community attribute. The community attribute is a group of destination addresses with the same characteristics and consists of a set of 4-byte values, each of which specifies a community. Generally, the community attribute on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X is in the format of *aa:nn*, where *aa* specifies a 2-byte AS number and *nn* specifies the community attribute ID defined by an administrator. The community attribute is not flexible enough because it fails to carry a 4-byte AS number and contains only one community attribute ID. To address this problem, the large-community attribute can be used instead. The large-community attribute consists of a set of 12-byte values and is in the format of *Global Administrator:LocalData1:LocalData2*.

 NOTE

The large-community filter applies only to BGP because the large-community attribute is also a private attribute of BGP. For details about the large-community attribute, see [Large-Community Attribute](#).

## Extended Community

An extended community is used to filter BGP routes based on extended community attributes. BGP extended community attributes are classified as follows:

- **VPN target:** A VPN target controls route learning between VPN instances, isolating routes of VPN instances from each other. A VPN target may be either an import or export VPN target. Before advertising a VPNv4 or VPNv6 route to a remote MP-BGP peer, a PE adds an export VPN target to the route. After receiving a VPNv4 or VPNv6 route, the remote MP-BGP peer compares the received export VPN target with the local import VPN target. If they are the same, the remote MP-BGP peer adds the route to the routing table of the local VPN instance.
- **Source of Origin (SoO):** Several CEs at a VPN site may be connected to different PEs. The VPN routes advertised from the CEs to the PEs may be re-advertised to the VPN site where the CEs reside after the routes have traversed the backbone network, causing routing loops at the VPN site. In this situation, configure an SoO attribute for VPN routes. With the SoO attribute, routes advertised from different VPN sites can be distinguished and will not be advertised to the source VPN site, preventing routing loops.
- **Encapsulation:** The encapsulation extended community attribute is classified as the VXLAN encapsulation extended community attribute or MPLS encapsulation extended community attribute. In EVPN VXLAN scenarios, EVPN routes carry the VXLAN encapsulation extended community attribute, and the value of this attribute can be set to 0:8 to filter EVPN routes. In EVPN MPLS scenarios, received EVPN routes do not carry the MPLS encapsulation extended community attribute in most cases. If a device receives EVPN routes with the MPLS encapsulation extended community attribute, the value of this attribute can be set to 0:10 to filter out the routes.
- **Segmented-nh:** The segmented-nh can be added to intra-AS I-PMSI A-D routes in an NG MVPN scenario where segmented tunnels are used.
- **Compress-algorithm:** In inter-chassis redirection scenarios for data compression and decompression, you can configure a compress-algorithm extended community filter and run the **if-match extcommunity-list compress-algorithm** command on the compressor. If the CPI value in the extended community attribute of a route is 0, the priority of the route is low.

The matching condition of an extended community can be specified using an extended community ID or a regular expression.

### NOTE

An extended community is used to filter only BGP routes because the extended community attribute is also a private attribute of BGP.

Regular expression matching is intensive processing of CPU computing. When a large number of regular expressions are configured in a policy to match a route attribute and the length of the route attribute is long, the processing performance of the policy deteriorates. To improve the processing performance of the route-policy, decrease the number of regular expressions or use non-regular expression matching commands.

## RD

An RD is used to filter BGP routes based on RDs in VPN routes. RDs are used to distinguish IPv4 and IPv6 prefixes in the same address space in VPN instances. RD-specific matching conditions can be configured in an RD filter.

For details about how to configure an RD, see *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Configuration Guide – VPN*.

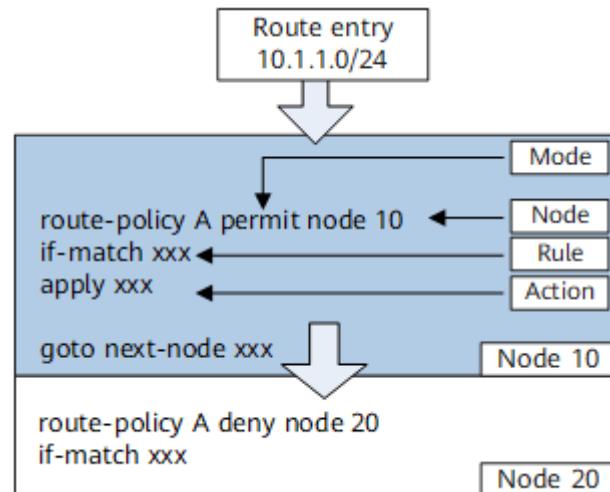
## Route-Policy

A route-policy is a complex filter. It is used to match attributes of specified routes and change route attributes when specific conditions are met. A route-policy can use the preceding six filters to define its matching rules.

### Composition of a Route-Policy

As shown in the following figure, a route-policy consists of node IDs, matching modes, **if-match** clauses, **apply** clauses, and **goto next-node** clauses. The **if-match**, **apply**, and **goto next-node** clauses are optional.

**Figure 1-464** Composition of a route-policy



### 1. Node ID

A route-policy can consist of multiple nodes. The method of specifying a node ID is the same as that of specifying an index for an IP prefix list. In a route-policy, routes are filtered based on the following rules:

- Sequential match: A device checks entries in ascending order by node ID. Specifying the node IDs in a required order is recommended.
- Unique match: The relationship among the nodes of a route-policy is "OR". If a route matches one node, the route matches the route-policy and will not be matched against a next node.

### 2. Matching mode

There are two matching modes: **permit** and **deny**.

- **permit**: indicates the permit mode of a node. If a route matches the **if-match** clauses of the node in permit mode, the **apply** clauses of the node are executed, and the route will not be matched against a next node. If the route does not match the **if-match** clauses of the node, the device continues to match the route against a next node.
- **deny**: indicates the deny mode of a node. In deny mode, **apply** clauses are not executed. If a route matches all **if-match** clauses of the node, the route is rejected and is not matched against a next node. If the entry does not match **if-match** clauses of the node, the device continues to match the route against a next node.

#### NOTE

To allow other routes to pass through, a route-policy that contains no **if-match** or **apply** clause in the permit mode is usually configured for a node after multiple nodes in the deny mode are configured.

### 3. **if-match** clause

The **if-match** clause defines the matching rules.

Each node of a route-policy can comprise multiple or none **if-match** clauses. By default, if the address family that a route belongs to does not match that specified in an if-match clause of a route-policy, the route matches the route-policy. Take a route-policy node in permit mode (permit node for short) as an example. If no if-match clause is configured for the permit node, all IPv4 and IPv6 routes are considered to match this node. If the permit node is configured with if-match clauses for filtering IPv4 routes only, IPv4 routes that match the if-match clauses and all IPv6 routes are considered to match this node. If the permit node is configured with if-match clauses for filtering IPv6 routes only, IPv6 routes that match the if-match clauses and all IPv4 routes are considered to match this node. This implementation also applies to a deny node.

You are not advised to use the same route-policy to filter both IPv4 and IPv6 routes by default. Otherwise, services may be interrupted in the following scenarios:

- For the same route-policy, some nodes apply only to IPv4 routes and some nodes apply only to IPv6 routes.
- A route-policy applies only to IPv4 routes but is used by IPv6 protocols.
- A route-policy applies only to IPv6 routes but is used by IPv4 protocols.

To use the same route-policy to filter both IPv4 and IPv6 routes, you can change the default behavior of the route-policy. When the address family that a route belongs to does not match that specified in an **if-match** clause of a route-policy, to set the default action of the route-policy to deny, run the **route-policy address-family mismatch-deny** command. Take a permit node as an example. If no **if-match** clause is configured for the permit node, all IPv4 and IPv6 routes are considered to match this node. If the permit node is configured with only an **if-match** clause for filtering IPv4 routes, only IPv4 routes that match the **if-match** clause are considered to match this node, and no IPv6 routes match this node. If the permit node is configured with only an **if-match** clause for filtering IPv6 routes, only IPv6 routes that match the **if-match** clause are considered to match this node, and no IPv4 routes match this node. This implementation also applies to a deny node.

 NOTE

If an **if-match** clause of a node uses information such as the next hop address or direct route source as a matching condition, the node compares the address family to which the next hop address or direct route source belongs with that specified in the **if-match** clause.

4. **apply** clause

The **apply** clauses specify actions. When a route matches a route-policy, the system sets some attributes for the route based on the **apply** clause.

Each node of a route-policy can comprise multiple **apply** clauses or no **apply** clause at all. No **apply** clause needs to be configured if routes are to be filtered but their attributes do not need to be set.

If **if-match** clauses are not configured in a route-policy but **apply** clauses are configured, the route-policy does not have any filtering conditions to match routes. In this case, if the matching mode of a route-policy node is set to **permit**, all routes are permitted and the **apply** clauses are executed; if the matching mode is set to **deny**, all routes are denied and the **apply** clauses are not executed.

5. **goto next-node** clause

**goto next-node** clauses further match routes against a specified node after the routes match the current node.

**Matching results of a route-policy**

The matching results of a route-policy are obtained based on the following aspects:

- Matching mode of the node, either **permit** or **deny**
- Matching rules (either permit or deny) contained in the **if-match** clause (using filters such as IP prefix lists or ACLs)

**Table 1-164** describes matching results.

**Table 1-164** Matching results of a route-policy

| Rule (Matching Rule Contained in if-match Clauses) | Mode (Matching Mode of a Node) | Matching Result                                                                                                                                                                                                                                                                                    |
|----------------------------------------------------|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| permit                                             | permit                         | <ul style="list-style-type: none"><li>• Routes matching the <b>if-match</b> clauses of the node match the route-policy, and the matching is complete.</li><li>• Routes not matching the <b>if-match</b> clauses of the node continue to match against the next node of the route-policy.</li></ul> |

| Rule (Matching Rule Contained in if-match Clauses) | Mode (Matching Mode of a Node) | Matching Result                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------------------------------------------|--------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                    | <b>deny</b>                    | <ul style="list-style-type: none"> <li>Routes matching the <b>if-match</b> clauses of the node are denied by the route-policy, and the matching is complete.</li> <li>Routes not matching the <b>if-match</b> clauses of the node continue to match against the next node of the route-policy.</li> </ul>                                                                                                                                                     |
| <b>deny</b>                                        | <b>permit</b>                  | <ul style="list-style-type: none"> <li>Routes matching the <b>if-match</b> clauses of the node are denied by the route-policy and continue to match against the next node.</li> <li>Routes not matching the <b>if-match</b> clauses of the node continue to match against the next node of the route-policy.</li> </ul>                                                                                                                                       |
|                                                    | <b>deny</b>                    | <ul style="list-style-type: none"> <li>Routes matching the <b>if-match</b> clauses of the node are denied by the route-policy and continue to match against the next node.</li> <li>Routes not matching the <b>if-match</b> clauses of the node continue to match against the next node of the route-policy.</li> </ul> <p><b>NOTE</b><br/>If all <b>if-match</b> clauses and nodes of the route-policy are in <b>deny</b> mode, all routes are rejected.</p> |

#### NOTE

By default, all the routes that do not match the filtering conditions in a route-policy on the HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series are rejected by the route-policy. If more than one node is defined in a route-policy, at least one of them must be in permit mode. The reason is as follows:

- If a route fails to match any of the nodes, the route is denied by the route-policy.
- If all the nodes in the route-policy are set in **deny** mode, all the routes to be filtered are denied by the route-policy.

## Other Functions

In addition to the preceding functions, routing policies have an enhanced feature: BGP to IGP.

In some scenarios, when an IGP uses a routing policy to import BGP routes, route attributes, the cost for example, can be set based on private attributes, such as the community in BGP routes. However, without the BGP to IGP feature, BGP routes

are denied because the IGP fails to identify private attributes, such as community attributes in these routes. As a result, **apply** clauses used to set route attributes do not take effect.

With the BGP to IGP feature, route attributes can be set based on private attributes, such as the community, extended community, and AS\_Path attributes in BGP routes. The BGP to IGP implementation process is as follows:

- When an IGP imports BGP routes through a route-policy, route attributes can be set based on private attributes, such as the community attribute in BGP routes.
- If BGP routes carry private attributes, such as community attributes, the system filters the BGP routes based on the private attributes. If the BGP routes meet the matching rules, the routes match the route-policy, and **apply** clauses take effect.
- If BGP routes do not carry private attributes, such as community attributes, the BGP routes fail to match the route-policy and are denied, and **apply** clauses do not take effect.

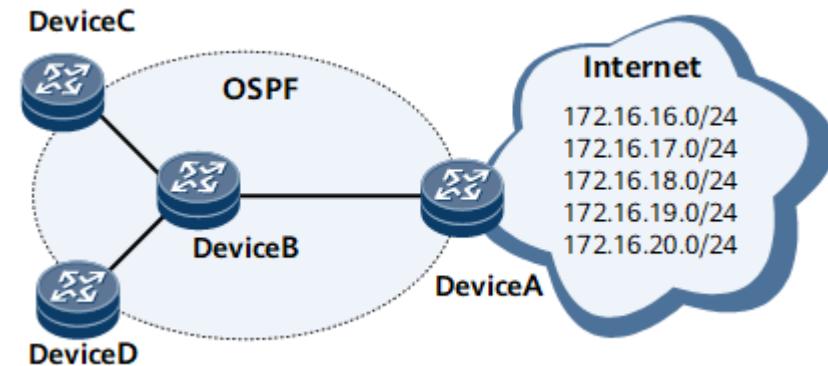
### 1.1.11.1.3 Application Scenarios for Routing Policies

#### Specific Route Filtering

On the OSPF-enabled network shown in [Figure 1-465](#), Device A receives routes from the Internet and advertises some of the routes to Device B.

- Device A advertises only routes 172.16.17.0/24, 172.16.18.0/24, and 172.16.19.0/24 to Device B.
- Device C accepts only the route 172.16.18.0/24.
- Device D accepts all the routes advertised by Device B.

[Figure 1-465](#) Filtering received and advertised routes



There are multiple approaches to meet the preceding requirements, and the following two approaches are used in this example:

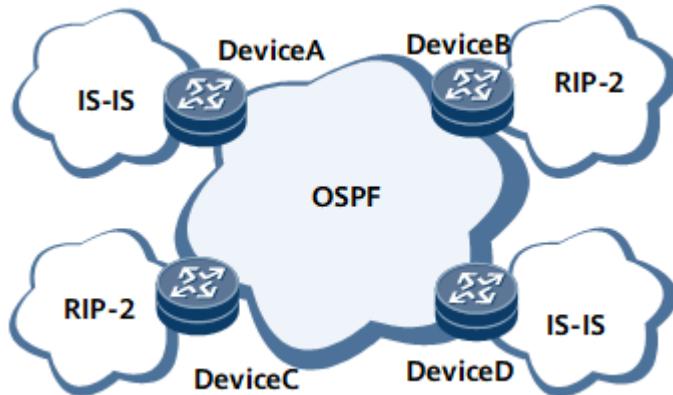
- Use IP prefix lists
  - Configure an IP prefix list for Device A and configure the IP prefix list as an export policy on Device A for OSPF.

- Configure another IP prefix list for Device C and configure the IP prefix list as an import policy on Device C for OSPF.
  - Use route-policies
    - Configure a route-policy (the matching rules can be the IP prefix list, route cost, or route tag) for Device A and configure the route-policy as an export policy on Device A for OSPF.
    - Configure another route-policy on Device C and configure the route-policy as an import policy on Device C for OSPF.
- Compared with an IP prefix list, a route-policy can change route attributes and control routes more flexibly, but it is more complex to configure.

## Applying a Routing Policy to Transparently Transmit Routes of Other Protocols Through OSPF

On the network shown in [Figure 1-466](#), an AS runs OSPF and functions as a transit AS for other areas. Routes from the IS-IS area connected to Device A need to be transparently transmitted through the OSPF AS to the IS-IS area connected to Device D. Routes from the RIP-2 area connected to Device B need to be transparently transmitted through the OSPF AS to the RIP-2 area connected to Device C.

**Figure 1-466** Transparently transmitting routes of other protocols through OSPF

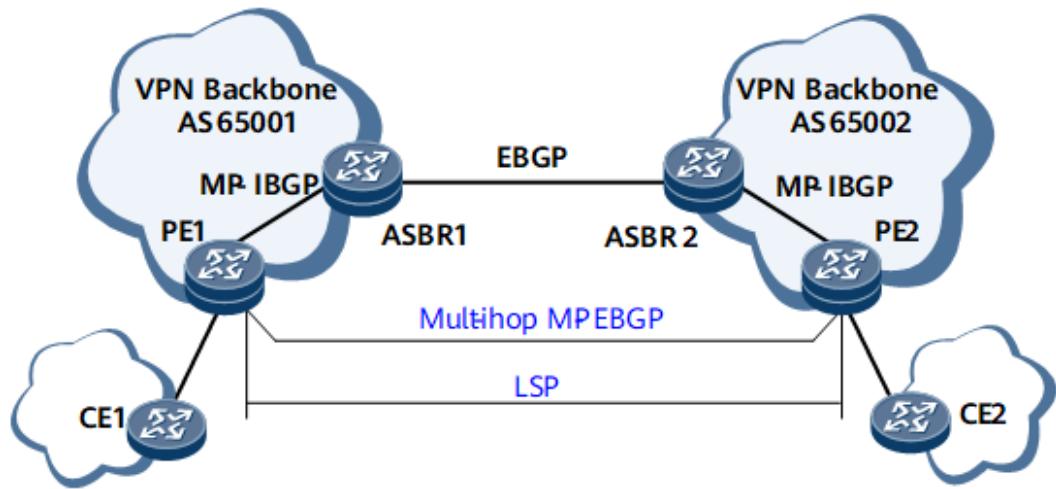


To meet the preceding requirements, configure a route-policy for Device A to set a tag for the imported IS-IS routes. Device D identifies the IS-IS routes from OSPF routes based on the tag.

## Applying a Routing Policy in Inter-AS VPN Option C

On the network shown in [Figure 1-467](#), CE1 and CE2 communicate with each other through inter-AS VPN Option C.

Figure 1-467 Implementing route-policies in the inter-AS VPN Option C scenario



To establish an inter-AS label switched path (LSP) between PE1 and PE2, route-policies need to be configured for autonomous system boundary routers (ASBRs).

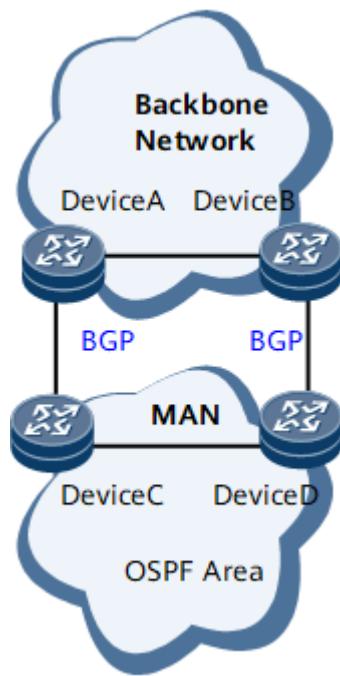
- When an ASBR advertises the routes received from a PE in the same AS to the peer ASBR, the ASBR allocates MPLS labels to the routes using a route-policy.
- When an ASBR advertises labeled IPv4 routes to a PE in the same AS, the ASBR reallocates MPLS labels to the routes using another route-policy.

In addition, to control route transmission between different VPN instances on a PE, configure a route-policy for the PE and configure the route-policy as an import or export policy for the VPN instances.

## Application of BGP to IGP

On the network shown in [Figure 1-468](#), Device A and Device B are aggregation devices on a backbone network, and Device C and Device D are egress devices of a metropolitan area network (MAN). BGP peer relationships are established between Device A and Device C as well as between Device B and Device D. External routes are advertised to the MAN using BGP. The MAN runs OSPF to implement interworking.

Figure 1-468 BGP to IGP



To enable devices on the MAN to access the backbone network, Device C and Device D need to import routes. When OSPF imports BGP routes, a routing policy can be configured to control the number of imported routes based on private attributes (such as the community) of the imported BGP routes or modify the cost of the imported routes to control the MAN egress traffic.

### 1.1.11.2 Routing Policy Configuration

Routing policies are applied to routing information to change the path through which network traffic passes.

#### 1.1.11.2.1 Overview of Routing Policies

Routing policies can be used to control route advertisement and acceptance.

#### Definition

Routing policies are used to filter routes and control how routes are received and advertised. If route attributes, such as reachability, are changed, the path along which network traffic passes changes accordingly.

#### Purpose

When advertising, receiving, and importing routes, the router implements certain routing policies based on actual networking requirements to filter routes and change the route attributes. Routing policies serve the following purposes:

- Control route advertising  
Only routes that match the rules specified in a policy are advertised.

- Control route receiving  
Only the required and valid routes are received, which reduces the routing table size and improves network security.
- Filter and control imported routes  
A routing protocol may import routes discovered by other routing protocols. Only routes that satisfy certain conditions are imported to meet the requirements of the protocol.
- Modify attributes of specified routes  
To enrich routing information, a routing protocol may import routing information discovered by other routing protocols. Only the routing information that satisfies the conditions is imported. Some attributes of the imported routing information are changed to meet the requirements of the routing protocol.

## Benefits

Routing policies have the following benefits:

- Control the routing table size, saving system resources.
- Control route receiving and advertising, improving network security.
- Modify attributes of routes for proper traffic planning, improving network performance.

## Differences Between the Routing Policy and Policy-based Routing

Unlike the routing mechanism that searches the forwarding table for matching routes based on the destination addresses of IP packets, policy-based routing (PBR) is based on the user-defined routing policies. PBR selects routes based on the user-defined routing policies, with reference to the source IP addresses and lengths of incoming packets. PBR can be used to improve security and implement load balancing.

A routing policy and PBR have different mechanisms. [Table 1-165](#) shows the differences between them.

**Table 1-165** Differences between the routing policy and PBR

| Routing Policy                                                           | Policy-based Routing                                                                                                                                           |
|--------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Forwards packets based on destination addresses in the routing table.    | Forwards packets based on the policy. The device searches the routing table for packet forwarding only after packets fail to be forwarded based on the policy. |
| Based on the control plane, serves routing protocols and routing tables. | Based on the forwarding plane, serves forwarding.                                                                                                              |
| Combines with a routing protocol to form a policy.                       | Needs to be configured hop by hop to ensure that packets are forwarded based on the policies.                                                                  |

| Routing Policy                                       | Policy-based Routing                                       |
|------------------------------------------------------|------------------------------------------------------------|
| Is configured using the <b>route-policy</b> command. | Is configured using the <b>policy-based-route</b> command. |

### 1.1.11.2.2 Configuration Precautions for Routing Policy

## Feature Requirements

Table 1-166 Feature requirements

| Feature Requirements                                                                                                                                                                                                                  | Series           | Models                                                                                     |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|--------------------------------------------------------------------------------------------|
| It is recommended that the number of nodes in each route-policy in the system be less than 1000. If the number exceeds 1000, the service processing performance of the route-policy will deteriorate and user experience is affected. | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| If more than one node is defined in a route-policy or another type of filter, at least one of them must be in permit mode; otherwise, all routes may be denied.                                                                       | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |
| If the route-policy nonexistent-config-check disable command is not run, nonexistent route-policies cannot be applied to services.                                                                                                    | NetEngine 8000 X | NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X |

### 1.1.11.2.3 Configuring an IP Prefix List

An IP prefix list filters routes based on the destination addresses of the routes.

## Usage Scenario

To control route reception and advertisement based on the destination addresses of routes, you need to configure an IP prefix list.

The address prefix is a mandatory attribute of all routes and can be flexibly used in various networking modes.

## Configuring an IP Prefix List

An IP prefix list matches IPv4 or IPv6 address prefixes, which are defined by an IP address and a mask length.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Configure an IP prefix list.

- To configure an IPv4 prefix list, run the **ip ip-prefix ip-prefix-name [ index index-number ] { permit | deny } ip-address mask-length [ match-network ] [ greater-equal greater-equal-value ] [ less-equal less-equal-value ]** command.

The mask length range can be expressed as *mask-length* <= *greater-equal-value* <= *less-equal-value* <= 32. If only **greater-equal** is specified, the prefix range is [*greater-equal-value*, 32]. If only **less-equal** is specified, the prefix range is [*mask-length*, *less-equal-value*].

An IPv4 prefix list is identified by its name, and each list contains one or multiple entries. Each entry is identified by an index, and can uniquely specify a matching range in the form of a network prefix. An IPv4 prefix list named **abcd** is used as an example.

```

ip ip-prefix abcd index 10 permit 10.0.0.0 8
ip ip-prefix abcd index 20 permit 10.1.0.0 16
```

During route matching, the system checks the entries identified by indexes in ascending order. If a route matches an entry, it is no longer matched against the next entry.

The IP prefix list on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X denies all unmatched routes by default. If all entries are in **deny** mode, all routes will be denied by the IP prefix list. In this case, after multiple entries are specified in **deny** mode, define an entry **permit 0.0.0.0 0 less-equal 32** to permit all the other IPv4 routes.

#### NOTE

If more than one prefix entry is defined, at least one of them must be in **permit** mode.

- To configure an IPv6 prefix list, run the **ip ipv6-prefix ipv6-prefix-name [ index index-number ] { permit | deny } ipv6-address prefix-length [ match-network ] [ greater-equal greater-equal-value ] [ less-equal less-equal-value ]** command.

An IPv6 prefix list is identified by its name, and each IPv6 prefix list contains one or multiple entries. Each entry can independently specify a matching range in the form of a network prefix and is identified by an index. An IPv6 prefix list named **abcd** is used as an example.

```

ip ipv6-prefix abcd index 10 permit 2001:db8:1:: 64
ip ipv6-prefix abcd index 20 permit 2001:db8:2:: 64
```

During route matching, the system checks the entries identified by indexes in ascending order. If a route matches an entry, the route is no longer matched against the next entry.

The IP prefix list on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X denies all unmatched routes by default. If all entries are in **deny** mode, all routes will be denied by the IP prefix list. In this case, after multiple entries are specified in deny mode, define an entry **permit :: 0 less-equal 128** to permit all the other IPv6 routes.

 **NOTE**

If more than one prefix entry is defined, at least one of them must be in **permit** mode.

**Step 3 Run commit**

The configuration is committed.

----End

## Verifying the Configuration

After configuring an IP prefix list, check its information.

## Prerequisites

The IP prefix list has been configured.

## Procedure

- Run the **display ip ip-prefix [ ip-prefix-name ]** command to check information about the IPv4 prefix list.
- Run the **display ip ipv6-prefix [ ipv6-prefix-name ]** command to check information about the IPv6 prefix list.

----End

### 1.1.11.2.4 Configuring an ACL

## Context

An ACL is a set of sequential filter rules. In an ACL rule, you can specify packet information, such as the inbound interface name, source or destination IP address, protocol type, source or destination port number, and permit or deny mode. Then, a device matches received packets against the rules and determines whether to accept or discard the packets.

An ACL consisting of a set of rules is used only to sort packets based on the defined rules. To filter packets, the ACL must be used together with a routing policy.

ACLs are classified as the ACLs that apply to IPv4 routes or those that apply to IPv6 routes. ACLs are also classified as interface-based ACLs, basic ACLs, or advanced ACLs based on the usage. You can specify an IP address and a subnet range in an ACL to match the source IP address, destination network segment address, or the next-hop address of each route.

ACLs can be configured on network devices, such as access and core devices, to improve network security and stability. ACLs can be used to provide the following functions:

- Protect devices against IP, TCP, and Internet Control Message Protocol (ICMP) packet attacks.
- Control network access. For example, ACLs can be used to control enterprise network users' access to external networks, the specific network resources that users can access, and the period during which users can access networks.
- Limit network traffic and improve network performance. For example, ACLs can be used to limit bandwidth for upstream and downstream traffic and to apply charging rules to user requested bandwidth, ensuring efficient utilization of network resources.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **route-policy route-policy-name { permit | deny } node node**

A route-policy node is created, and the route-policy view is displayed.

### Step 3 Run **if-match acl { acl-number| acl-name }**

An ACL-based matching rule is defined.

### Step 4 Run **quit**

Return to the system view.

### Step 5 Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The ACL view is displayed.

### Step 6 Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spec-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule is configured for the ACL.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

#### Step 7 Run **commit**

The configuration is committed.

----End

### Verifying the Configuration

Run the **display acl** command to check the rules of the configured ACL.

#### 1.1.11.2.5 Configuring an AS\_Path Filter

### Context

An AS\_Path filter is used to filter BGP routes by matching against AS\_Path attributes. The AS\_Path attribute records the numbers of all ASs through which a BGP route passes from the local end to the destination in the distance-vector (DV) order. In this case, AS\_Path attribute-based filtering rules can be defined to filter BGP routes.

The matching condition of an AS\_Path filter is specified using a regular expression. For example, ^30 indicates that only the AS\_Path attribute starting with 30 is matched. Using a regular expression can simplify configurations. For details on how to use regular expressions, see the command line description in the *Configuration Guide > Basic Configuration*.

#### NOTE

The AS\_Path filter takes effect only on BGP routes because the AS\_Path attribute is a private attribute of BGP.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **ip as-path-filter { as-path-filter-number| as-path-filter-name } [ index index-number ] { permit | deny } regular-expression**

An AS\_Path filter is configured.

*regular-expression* defines a matching rule for the AS\_Path filter.

#### Step 3 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the **display ip as-path-filter [ as-path-filter-number | as-path-filter-name ]** command to check information about the configured AS\_Path filter.

### 1.1.11.2.6 Configuring a Community Filter

#### Context

A Community filter is used to filter BGP routes based on Community attributes contained in the BGP routes. A Community attribute is a set of destination addresses with the same characteristics. Filtering rules defined based on Community attributes can be used to filter BGP routes.

In addition to the well-known Community attributes, users can define community attributes using digits. The matching condition of a community filter can be specified using a community ID or a regular expression.



Like an AS\_Path attribute, a Community attribute is used to filter only BGP routes because the Community attribute is also a private attribute of BGP.

#### Procedure

##### Step 1 Run system-view

The system view is displayed.

##### Step 2 Run ip community-filter

A community filter is configured.

- To configure a basic community filter, run the **ip community-filter basic comm-filter-name [ index index-number ] { permit | deny } [ community-number | aa:nn | internet [ strict-match ] | no-export-subconfed | no-advertise | no-export ] &<1-20>** or **ip community-filter basic-comm-filter-num [ index index-number ] { permit | deny } [ community-number | aa:nn | internet | no-export-subconfed | no-advertise | no-export ] &<1-20>** command.
- To configure an advanced community filter, run the **ip community-filter { advanced comm-filter-name | adv-comm-filter-num } [ index index-number ] { permit | deny } regular-expression** command.

##### Step 3 Run commit

The configuration is committed.

----End

## Verifying the Configuration

Run the **display ip community-filter [ basic-comm-filter-num | adv-comm-filter-num | comm-filter-name ]** command to check information about the configured Community filter.

### 1.1.11.2.7 Configuring a Large-Community Filter

#### Context

A Large-Community filter is used to filter BGP routes based on Large-Community attributes contained in BGP routes. The Large-Community attribute is an extended Community attribute. The Community attribute is a group of destination addresses with the same characteristics and consists of a set of 4-byte values, each of which specifies a community. Generally, the Community attribute on the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X is in the format of *aa:nn*, where *aa* specifies a 2-byte AS number and *nn* specifies the Community attribute ID defined by an administrator. The Community attribute is not flexible enough because it fails to carry a 4-byte AS number and contains only one Community attribute ID. To address this problem, the Large-Community attribute can be used instead. The Large-Community attribute consists of a set of 12-byte values and is in the format of *Global Administrator:LocalData1:LocalData2*.



The Large-Community filter is used to filter only BGP routes because it is a private BGP attribute.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run either of the following commands as needed:

- To configure a basic Large-Community filter, run the **ip large-community-filter basic *large-comm-filter-name* [ index *index-number* ] { permit | deny } { aa:bb:cc } &<1-16>** command.
- To configure an advanced Large-Community filter, run the **ip large-community-filter advanced *large-comm-filter-name* [ index *index-number* ] { permit | deny } regular-expression** command.

##### Step 3 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the **display ip large-community-filter [ *large-comm-filter-num* ]** command to check information about the configured Large-Community filter.

### 1.1.11.2.8 Configuring an Extcommunity Filter

#### Context

An extended community (extcommunity) filter is used to filter BGP routes based on extcommunity attributes. BGP extcommunity attributes are classified as follows:

- VPN target: A VPN target controls route learning between VPN instances, isolating routes of VPN instances from each other. VPN targets include export and import ones. Before advertising Virtual Private Network version 4 (VPNv4) or Virtual Private Network version 6 (VPNv6) routes to a remote Multiprotocol Extensions for Border Gateway Protocol (MP-BGP) peer, a PE adds export VPN targets to the routes. After receiving the VPNv4 or VPNv6 routes, the remote MP-BGP peer determines which routes can be added to its local VPN instance routing table based on whether the export VPN targets carried in the routes match the import VPN target of the local VPN instance.
- Source of Origin (SoO): Several CEs at a VPN site may be connected to different PEs. Routes advertised from the CEs to the PEs may be advertised back to the VPN site after the routes traverse the VPN backbone network. This may cause routing loops at the VPN site. To prevent routing loops, SoO attributes can be configured for routes from different VPN sites for differentiation.
- Encapsulation: The encapsulation extcommunity attribute is classified as the VXLAN encapsulation extcommunity attribute or MPLS encapsulation extcommunity attribute. In EVPN VXLAN scenarios, EVPN routes carry the VXLAN encapsulation extcommunity attribute, and the value of this attribute can be set to 0:8 to filter EVPN routes. In EVPN MPLS scenarios, received EVPN routes do not carry the MPLS encapsulation extcommunity attribute in most cases. If a device receives EVPN routes carrying the MPLS encapsulation extcommunity attribute, the value of this attribute can be set to 0:10 to filter these routes.
- Segmented-nh: The segmented-nh extcommunity attribute can be added to intra-AS I-PMSI A-D routes in an NG MVPN scenario where segmented tunnels are used.
- Compress-algorithm: In inter-chassis redirection scenarios, you can configure a compress-algorithm extended community filter and run the **if-match extcommunity-list compress-algorithm** command on the compressor. If the CPI value in the extended community attribute of a route is 0, the priority of the route is low.

The matching condition of an extcommunity filter can be specified using an extcommunity ID or a regular expression.



An extcommunity filter is used to filter only BGP routes because the extcommunity attribute is also a private attribute of BGP.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

**Step 2** Configure the following extcommunity filters as needed.

Configure a VPN-Target extcommunity filter:

- To configure a basic VPN-Target extcommunity filter, run the **ip extcommunity-filter { basic-extcomm-filter-num | basic basic-extcomm-filter-name } [ index index-number ] { deny | permit } { rt { as-number:nn | 4as-number:nn | ipv4-address.nn } } &<1-16>** command.
- To configure an advanced VPN-Target extcommunity filter, run the **ip extcommunity-filter { advanced-extcomm-filter-num | advanced advanced-extcomm-filter-name }[ index index-number ] { deny | permit } regular-expression** command.

Configure an SoO extcommunity filter:

- To configure a basic SoO extcommunity filter, run the **ip extcommunity-list soo basic basic-extcomm-filter-name [ index index-number ] { permit | deny } { site-of-origin } &<1-16>** command.
- To configure an advanced SoO extcommunity filter, run the **ip extcommunity-list soo advanced advanced-extcomm-filter-name [ index index-number ] { permit | deny } regular-expression** command.

Configure an encapsulation extcommunity filter:

- To configure a basic encapsulation extcommunity filter, run the **ip extcommunity-list encapsulation basic encapsulation-name [ index index-number ] { permit | deny } { encapsulation-value } &<1-16>** command.
- To configure an advanced encapsulation extcommunity filter, run the **ip extcommunity-list encapsulation advanced encapsulation-name [ index index-number ] { permit | deny } regular** command.

Configure a segmented-nh extcommunity filter:

- To configure a basic segmented-nh extcommunity filter, run the **ip extcommunity-list segmented-nh basic segmented-nh-name [ index index-number ] { permit | deny } { segmented-nh-value } &<1-16>** command.
- To configure an advanced segmented-nh extcommunity filter, run the **ip extcommunity-list segmented-nh advanced segmented-nh-name [ index index-number ] { permit | deny } regular** command.

Configure a compress-algorithm extended community attribute:

- To configure a basic compress-algorithm extended community filter, run the **ip extcommunity-list compress-algorithm basic basicCompressName [ index index-val ] matchMode>{ extCmntyStr } &<1-16>** command.

Multiple entries (or rules) can be defined in an extcommunity filter, and the relationship between them is OR, which means that the route matches the extcommunity filter if it matches one of the rules.

**Step 3** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

- Run the **display ip extcommunity-filter** command to check information about the configured extended community filters.
- Run the **display ip extcommunity-list soo [ eclSooName ]** command to check detailed information about a configured SoO extended community filter.
- Run the **display ip extcommunity-list encapsulation [ name ]** command to check detailed information about a configured encapsulation extended community filter.
- Run the **display ip extcommunity-list segmented-nh [ eclSnhName ]** command to check detailed information about a configured segmented-nh extended community filter.
- Run the **display ip extcommunity-list compress-algorithm [ compressName ]** command to check detailed information about a configured compress-algorithm extended community filter.

### 1.1.11.2.9 Configuring an RD Filter

#### Context

An RD filter is used to filter routes based on route distinguishers (RDs) contained in VPN routes. RDs are used to distinguish IPv4 or IPv6 prefixes within the same address segment in a VPN instance. RD-specific matching conditions can be configured in an RD filter.

#### Procedure

##### Step 1 Run **system-view**

The system view is displayed.

##### Step 2 Run **ip rd-filter rdfIndex [ index index-number ] matchMode rdStr &<1-10>**

An RD filter is configured.

##### Step 3 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

Run the **display ip rd-filter [ rd-filter-number ]** command to check information about the configured RD filter.

### 1.1.11.2.10 Configuring a Route-Policy

Each node of a route-policy can comprise a set of **if-match**, **apply**, and **goto next-node** clauses.

## Usage Scenario

A route-policy includes various matching rules and hence can meet the requirements of various scenarios. Except ACLs, IP prefix lists, and AS\_Path filters, other filters need to be used with a route-policy.

A route-policy is used to match routes or route attributes, and to change route attributes when the matching rules are met. ACLs, IP prefix lists, AS\_Path filters, community filters, extended community filters, and RD filters can be used to define matching conditions.

A route-policy can consist of multiple nodes, and each node can comprise the following clauses:

- **if-match** clauses: define the matching rules that are used to match certain route attributes. The matching rules are conditions defined by the route-policy against which routes are matched.
- **apply** clauses: specify actions. When a route matches a node, the **apply** clauses set certain attributes for the route.
- **goto next-node** clauses: further match routes against a specified node after the routes match the current node.

For more information about a route-policy, refer to the *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description - IP Routing*.

## Pre-configuration Tasks

Before configuring a route-policy, complete the following tasks:

- [Configure an IP prefix list](#).
- Configure a routing protocol.

## Creating a Route-Policy

By applying a route-policy, you can set attributes for the imported routes as required.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **route-policy route-policy-name { permit | deny } node node**

A route-policy node is created, and the route-policy view is displayed.

Either of the following parameters can be used to define a matching mode:

- **permit**: configures the permit mode for the node. If a route matches the **if-match** clauses of the node in permit mode, the **apply** clauses of the node are executed, and the route will not be matched against a next node. If the route does not match the **if-match** clauses of the node, the device continues to match the route against a next node.

- **deny:** configures the deny mode for the node. In deny mode, **apply** clauses are not executed. If a route matches all **if-match** clauses of the node in deny mode, the route is rejected and is not matched against a next node. If the route does not match **if-match** clauses of the node, the device continues to match the route against a next node.

#### NOTE

By default, the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X considers that each route that does not match a route-policy is rejected by the route-policy. If more than one node is defined in a route-policy, at least one should be set to permit mode.

When a route-policy is used to filter routes, if a route does not match the matching rules of all nodes in the route-policy, the route is rejected by the route-policy. If all nodes in a route-policy are in deny mode, all routes are rejected by the route-policy.

When a route-policy is used to filter routes, the node with the smallest node ID is matched against first.

#### Step 3 (Optional) Run **quit**

Return to the system view.

#### Step 4 (Optional) Run **route-policy route-policy-name address-family mismatch-deny**

The device is configured to deny routes if the address family that the routes belong to does not match that specified in an **if-match** clause of the route-policy.

By default, if the address family that a route belongs to does not match that specified in an **if-match** clause of a route-policy, the route matches the route-policy. Take a route-policy node in permit mode (permit node for short) as an example. If no **if-match** clause is configured for the permit node, all IPv4 and IPv6 routes are considered to match this node. If the permit node is configured with **if-match** clauses for filtering IPv4 routes only, IPv4 routes that match the **if-match** clauses and all IPv6 routes are considered to match this node. If the permit node is configured with **if-match** clauses for filtering IPv6 routes only, IPv6 routes that match the **if-match** clauses and all IPv4 routes are considered to match this node. This implementation also applies to a deny node. When the default configuration is used, you are not advised to use the same route-policy to filter both IPv4 and IPv6 routes. Otherwise, services may be interrupted.

If you want to use the same route-policy to filter both IPv4 and IPv6 routes, you can run the **route-policy address-family mismatch-deny** command to change the default behavior of the route-policy in order to prevent a potential service interruption. After this configuration completes, if the address family that a route belongs to does not match that specified in an **if-match** clause of the route-policy, the route fails to match the route-policy. Take a permit node as an example. If no **if-match** clause is configured for the permit node, all IPv4 and IPv6 routes are considered to match this node. If the permit node is configured with **if-match** clauses for filtering IPv4 routes only, only IPv4 routes that match the **if-match** clauses are considered to match this node, and no IPv6 routes match this node. If the permit node is configured with **if-match** clauses for filtering IPv6 routes only, only IPv6 routes that match the **if-match** clauses are considered to match this node, and no IPv4 routes match this node. This implementation also applies to a deny node.

### Step 5 Run commit

The configuration is committed.

----End

## (Optional) Configuring an if-match Clause

The **if-match** clauses define the matching rules that are used to match certain route attributes.

## Procedure

### Step 1 Run system-view

The system view is displayed.

### Step 2 Run **route-policy route-policy-name { permit | deny } node node**

The route-policy view is displayed.

### Step 3 Configure **if-match** clauses in the route-policy.

- Based on a basic ACL:
  - a. Run the **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]** command to enter the ACL view.
  - b. Run the **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \*** command to configure a rule for the ACL.
  - c. Run the **if-match acl { acl-number| acl-name }** command to configure an ACL-based matching rule.
  - d. Run the **quit** command to return to the system view.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes.

Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- To configure a rule to match routes against a specified cost, run the **if-match cost cost** or **if-match cost { greater-equal greater-equal-value [ less-equal less-equal-value ] | less-equal less-equal-value }** command.
- To configure a rule to match routes against a specified outbound interface, run the **if-match interface { { interface-name | interface-type interface-number } &<1-16> }** command.
- To configure a rule to match routes against a specified route preference, run the **if-match preference preference** command.
- To configure a rule to match IPv4 routes against a specified next hop or source address, run the **if-match ip { next-hop | route-source | group-address } { acl { acl-number | acl-name } | ip-prefix ip-prefix-name }** command.
- To configure a rule to match routes against a specified IP prefix list, run the **if-match ip-prefix ip-prefix-name** command.
- To configure a rule to IPv6 routes, run the **if-match ipv6 { address | next-hop | route-source } { acl { acl-number | acl-name } | prefix-list ipv6-prefix-name }** command.
- Run any of the following commands as needed to match routes of a specific type:
  - To configure a rule to match OSPF routes of a specified type, run the **if-match route-type { external-type1 | external-type1or2 | external-type2 | internal | nssa-external-type1 | nssa-external-type1or2 | nssa-external-type2 }** command.
  - To configure a rule to match IS-IS routes of a specified level, run the **if-match route-type { is-is-level-1 | is-is-level-2 }** command.
  - To configure a rule to match BGP routes, run the **if-match route-type { ibgp | ebpg }** command.
  - To configure a rule to match MVPN routes, run the **if-match route-type mvpn { 1 | 3 } \*** command.
  - To configure a rule to match EVPN routes, run the **if-match route-type evpn { ad | es | inclusive | mac | prefix | join | leave | smet } \*** command.
  - To match BGP-LS routes, run the **if-match route-type bgp-ls { node | link | ipv4-prefix | ipv6-prefix } \*** command.
- To configure a rule to match routes carrying a specified tag value, run the **if-match tag tag** command.
- To configure a rule to match routes of a specific type of protocol, run the **if-match protocol { direct | static | rip | ripng | ospf | ospfv3 | bgp | isis | unr } \*** command.
- To configure a rule to match routes against a specified AS\_Path length, run the **if-match as-path length** command.

- To configure a rule to match routes against a specified AS\_Path filter, run the **if-match as-path-filter** command.
- To configure a rule to match routes against a specified Community filter, run the **if-match community-filter** command.
- To configure a rule to match routes against a specified extcommunity filter, run any of the following commands as needed:
  - To match routes against a VPN target extcommunity filter, run the **if-match extcommunity-filter** command.
  - To match routes against an encapsulation extcommunity filter, run the **if-match extcommunity-list encapsulation *encapsulation-name*** command.
  - To match routes against an SoO extcommunity filter, run the **if-match extcommunity-list soo *extcomm-filter-name*** command.
  - To match routes against a segmented-nh extcommunity filter, run the **if-match extcommunity-list segmented-nh *segmented-nh-name*** command.
  - To match the compress-algorithm extended community attribute list, run the **if-match extcommunity-list compress-algorithm *compress-name*** command.
- To configure a rule to match routes against a specified Large-Community filter, run the **if-match large-community-filter *lcfName* [ whole-match ]** command.
- To configure a rule to match routes against a specified RD filter, run the **if-match rd-filter *rd-filter-number*** command.
- To configure a rule to match BGP routes against a specified origin AS validation result, run the **if-match rpk1 origin-as-validation { valid | invalid | not-found }** command.
- To configure a rule to match routes against a specified MPLS label, run the **if-match mpls-label** command.
- To configure a rule to match routes against a specified MPLS Label2 value, run the **if-match mpls-label2** command.
- To configure a rule to match routes against a specified Layer 2 VNI list, run the **if-match l2vni [ *l2vni-list-name* ]** command.
- To configure a rule to match routes against a specified Layer 3 VNI list, run the **if-match l3vni [ *l3vni-list-name* ]** command.
- To configure a rule to match routes against a specified MAC address list, run the **if-match mac-list *mac-list-name*** command.
- To configure a rule to match routes against a specified Ethernet tag list, run the **if-match eth-tag-list *eth-tag-list-name*** command.
- To configure a rule to match routes against a multicast address, run the **if-match ip group-address { acl { *acl-number* | *acl-name* } | ip-prefix *ip-prefix-name* }** command.
- To configure a rule to match routes against the IP address of a route advertiser, run the **if-match ip route-originator { ip-prefix *ip-prefix-name* | acl { *acl-number* | *acl-name* } }** command.
- To match non-optimal BGP routes, run the **if-match route-state bgp-not-best** command.

- To configure a rule to match the source IP address in the BGP Flow Specification route prefix, run the **if-match flowspec source ip-prefix *ip-prefix-name*** command.
- To configure a rule to match the source IPv6 address in the BGP Flow Specification route prefix, run the **if-match flowspec source ipv6 prefix-list *ipv6-prefix-name*** command.
- To configure a rule to match the destination IP address in the BGP Flow Specification route prefix, run the **if-match flowspec destination ip-prefix *ip-prefix-name*** command.
- To configure a rule to match the destination IPv6 address in the BGP Flow Specification route prefix, run the **if-match flowspec destination ipv6 prefix-list *ipv6-prefix-name*** command.

The commands in Step 3 can be run in any required order. A node may have multiple **if-match** clauses or no **if-match** clause.

 NOTE

If multiple **if-match** clauses of a node in a route-policy define the same matching condition type, the relationship between them is "OR"; if the **if-match** clauses define different matching condition types, the relationship between these clauses is "AND". If you run any of the following **if-match** commands more than once, the latest configuration overrides the previous one:

- **if-match acl { acl-number | acl-name }**
- **if-match cost cost**
- **if-match extcommunity-list soo extcomm-filter-name**
- **if-match ip next-hop { acl { acl-number | acl-name } | ip-prefix ip-prefix-name }**
- **if-match ip route-source { acl { acl-number | acl-name } | ip-prefix ip-prefix-name }**
- **if-match ip group-address { acl { acl-number | acl-name } | ip-prefix ip-prefix-name }**
- **if-match ip route-originator { ip-prefix ip-prefix-name | acl { acl-number | acl-name } }**
- **if-match ip-prefix ip-prefix-name**
- **if-match ipv6 address { acl { acl-number | acl-name } | prefix-list ipv6-prefix-name }**
- **if-match ipv6 next-hop { acl { acl-number | acl-name } | prefix-list ipv6-prefix-name }**
- **if-match ipv6 route-source { acl { acl-number | acl-name } | prefix-list prefix-list ipv6-prefix-name }**
- **if-match rd-filter rd-filter-number**
- **if-match rpk1 origin-as-validation { invalid | not-found | valid }**
- **if-match tag tag**
- **if-match preference preference**
- **if-match protocol { direct | static | rip | ripng | ospf | ospfv3 | bgp | isis | unr } \***
- **if-match { mpls-label | mpls-label2 } \***
- **if-match as-path length**
- **if-match l2vni [ l2vni-list-name ]**
- **if-match l3vni [ l3vni-list-name ]**
- **if-match mac-list mac-list-name**
- **if-match eth-tag-list eth-tag-list-name**
- **if-match extcommunity-list encapsulation encapsulation-name**
- **if-match extcommunity-list segmented-nh segmented-nh-name**
- **if-match extcommunity-list compress-algorithm compress-name**
- **if-match route-state bgp-not-best**
- **if-match flowspec source ip-prefix ip-prefix-name**
- **if-match flowspec source ipv6 prefix-list ipv6-prefix-name**
- **if-match flowspec destination ip-prefix ip-prefix-name**
- **if-match flowspec destination ipv6 prefix-list ipv6-prefix-name**

If no **if-match** clause is specified, all routes will match the route-policy node.

You are advised not to use the same route-policy to filter both IPv4 and IPv6 routes when the **route-policy address-family mismatch-deny** command is not configured. Otherwise, services may be interrupted in the following scenarios:

1. For the same **route-policy**, some nodes match IPv4 routes, and some nodes match IPv6 routes.
2. A **route-policy** matches only IPv4 routes, but the **route-policy** is referenced by IPv6.

3. A **route-policy** matches only IPv6 routes, but the **route-policy** is referenced by IPv4.

#### Step 4 Run commit

The configuration is committed.

----End

### (Optional) Configuring an apply Clause

The **apply** clauses specify actions to set certain route attributes.

## Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run **route-policy route-policy-name { permit | deny } node node**

The route-policy view is displayed.

#### Step 3 Run the following commands as needed to configure **apply** clauses for the route-policy:

- To set an AIGP value for matched BGP routes, run the **apply aigp { [ + | - ] cost | inherit-cost }** command.
- To set a cost for matched routes, run the **apply cost { [ + | - ] cost | inherit | none }** command.
- To set the AS\_Path attribute, run the **apply as-path { as-number-plain | as-number-dot } &<1-128>{ additive | overwrite | delete }** or **apply as-path asValues { additive | overwrite | delete }** command.
- To clear the original AS\_Path attribute, run the **apply as-path none overwrite** command.
- To set the number of times the most recent AS number in AS\_Path is appended to routes, run the **apply as-path most-recent <most-recent-value>** command.
- To set a tag for a multi-homed inbound interface group (MIIG) on a source address validation network (SAVNET), run the **apply savnet miig-tag tagVal** command.

#### NOTE

The NetEngine 8100 X8 does not support the configuration of a tag for an MIIG on a SAVNET.

- To delete a community attribute from BGP routes based on the specified value in a community filter, run the **apply comm-filter { basIndex | advIndex } delete** or **apply comm-filter cmntyName delete** command.
- To set a BGP community attribute, run the **apply community { cmntyValue | cmntyNum | internet | no-advertise | no-export | no-export-subconfed } &<1-32> [ additive ]** or **apply community community-list community-list-name [ additive ]** command.
- Set an extended community attribute for BGP routes.
  - To set the VPN target extcommunity attribute, run the **apply extcommunity { rt extCmntyValue } &<1-16> [ additive ]** command.

- To set the color extcommunity attribute, run the **apply extcommunity color extCmntyValue** command.
- To set the SoO extended community attribute, run the **apply extcommunity soo { site-of-origin } &<1-16> additive** command.
- To set the bandwidth extended community attribute, run the **apply extcommunity bandwidth { extCmntyString | none }**, **apply extcommunity bandwidth aggregate-upstream [ limit upstream-bandwidth ]**, or **apply extcommunity bandwidth aggregate [ limit bandwidth-value ]** command.
- To clear the existing extended community attributes of routes, run the **apply extcommunity rt none** command.
- To set the redirection extended community attribute for matched BGP Flow Specification routes, run the **apply extcommunity redirect ip ipv4-address:nn** or **apply extcommunity redirect vpn-target extCmntyValue** command.
- To set the Large-Community attribute of BGP routes, run the **apply large-community { aa:bb:cc } &<1-16> { additive | overwrite | delete }** or **apply large-community-list large-community-list-name { additive | overwrite | delete }** command.
- To set the Local\_Pref attribute for BGP routes, run the **apply local-preference [ + | - ] localPreference** command.
- To allocate MPLS labels to public network routes, run the **apply mpls-label** command.
- To set the QoS parameter **ip-precedence** for matched routes, run the **apply ip-precedence ip-precedence** command.
- To set the local QoS ID for matched routes, run the **apply qos-local-id qos-local-id** command.
- To set the Origin attribute of BGP routes, run the **apply origin { egp { egpVal } | igrp | incomplete }** command.
- To set the PrefVal attribute of BGP routes, run the **apply preferred-value preferredVal** command.
- To set the BGP traffic index for statistics collection, run the **apply traffic-index indexVal** command.
- To set a cost type for matched routes, run the **apply cost-type { external | internal | type-1 | type-2 | internal-inc-ibgp | med-plus-igp | med-inherit-aigp }** command.
- To set dampening parameters for EBGP routes, run the **apply dampening half-life-reach reuse suppress ceiling** command.
- To set a next-hop address for IPv4 routes, run the **apply ip-address next-hop { ipv4-address | peer-address | blackhole | original }** command.
- To set a next-hop address for IPv6 routes, run the **apply ipv6 next-hop { ipv6-address | peer-address | blackhole | original }** command.
- To set a level for IS-IS routes, run the **apply isis { level-1 | level-1-2 | level-2 }** command.
- To import routes to a specified OSPF area, run the **apply ospf { backbone | stub-area }** command.
- To set the preference of protocol routes, run the **apply preference preference** command.

- To set a tag value for matched routes, run the **apply tag tag** command.
- To set a peer group ID, run the **apply peer-id peerId** command.
- To set a gateway IP address for matched routes, run the **apply gateway-ip { origin-nexthop | ip-address | none }** command.
- To set a gateway IPv6 address for matched routes, run the **apply ipv6 gateway-ip { origin-nexthop | ipv6-address | none }** command.
- To discard entropy labels of matched routes, run the **apply entropy-label none** command.
- To stitch PMSIs, run the **apply stitch-pmsi { incoming { rsvp-te p2mp-template p2mp-template-name [ mldp root-ip root-ip-address ] } | mldp [ root-ip root-ip-address ] | rsvp-te p2mp-template p2mp-template-name }** command.
- To set a Flex-Algo ID for matched routes, run the **apply flex-algo flex-algo-id** command.

 **NOTE**

The commands in Step 3 can be run in any required order. A node can have multiple or no **apply** clauses.

**Step 4 Run commit**

The configuration is committed.

----End

**(Optional) Further Matching Routes Against a Specified Node**

A route-policy can be configured to match routes against two or more nodes.

**Context**

The relationship between the matching rules of nodes in the same route-policy is OR. Specifically, if a route matches a node, it matches the route-policy and is no longer matched against other nodes. If you need to match the route against two or more nodes, configure a route-policy and use it to match the route against a specified node after the route matches the current node.

**Procedure**

**Step 1 Run system-view**

The system view is displayed.

**Step 2 Run route-policy route-policy-name { permit | deny } node node**

The route-policy view is displayed.

**Step 3 Run goto next-node [ node ]**

The route-policy is configured to further match routes against a specified node after the routes match the current node.

If *node* is not specified in the command, the route will be further matched against the next node of the current node by default.

If the *node* specified in the command does not exist, the route will be further matched against the next node of the specified node by default. If the next node of the specified node does not exist either, the route fails to match the route-policy, and no **apply** clause will be applied to the route.

#### Step 4 Run **commit**

The configuration is committed.

----End

### Applying a Route-Policy

A route-policy takes effect only when it is applied to a routing protocol.

### Context

Route-policies apply to direct routes, static routes, RIP/RIPng, IS-IS, OSPF/OSPFv3, BGP/MPLS IP VPN, and BGP/BGP4+. In addition, a route-policy can be applied to an FRR scenario. Choose one of the following configurations as needed:

- [Apply a route-policy to direct routes.](#)
- [Apply a route-policy to static routes.](#)
- [Apply a route-policy to IPv6 static routes.](#)
- [Apply a route-policy to RIP routes.](#)
- [Apply a route-policy to RIPng routes.](#)
- [Apply a route-policy to IPv4 IS-IS routes.](#)
- [Apply a route-policy to IPv6 IS-IS routes.](#)
- [Apply a route-policy to OSPF routes.](#)
- [Apply a route-policy to OSPFv3 routes.](#)
- [Apply a route-policy to BGP routes.](#)
- [Apply a route-policy to BGP4+ routes.](#)
- [Apply a route-policy to BGP/MPLS IP VPN routes.](#)
- Apply a route-policy to EVPN.

### Procedure

- Apply a route-policy to direct routes.
  - a. Run **system-view**  
The system view is displayed.
  - b. Perform the following operations as required to apply a route-policy to direct routes. For details, see [Table 1-167](#).

**Table 1-167** Applying a route-policy to direct routes

| Objectives                                                                                                                                                                                                              | Command                                                                               | Reference                                                                                                      |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------|
| To configure the device to advertise the public network ARP Vlink direct routes that match a route-policy (specified by the <b>route-policy</b> <i>route-policy-name</i> parameter)                                     | <b>arp vlink-direct-route advertise [ route-policy <i>route-policy-name</i> ]</b>     | <a href="#">1.1.1.2.16 Configuring the Advertisement of Public Network IPv4 ARP Vlink Direct Routes</a>        |
| To configure a device to advertise IPv6 Neighbor Discovery Protocol (NDP) Vlink direct routes that match a route-policy (specified by the <b>route-policy</b> <i>route-policy-name</i> parameter) on the public network | <b>ipv6 nd vlink-direct-route advertise [ route-policy <i>route-policy-name</i> ]</b> | <a href="#">1.1.1.2.17 Configuring the Advertisement of IPv6 NDP Vlink Direct Routes on the Public Network</a> |

| Objectives                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | Command                                                              | Reference |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|-----------|
| <p>To apply a route-policy to direct routes on the public network</p> <p><b>NOTE</b></p> <p>Currently, QoS information, such as the traffic behavior or local ID, can be configured for direct routes only based on a route-policy. A router applies different QoS policies to packets based on the QoS information while forwarding these packets. In this manner, traffic statistics can be collected, and authentication and accounting can be performed based on the direct routes.</p> | <b>ip direct-routing-table route-policy <i>route-policy-name</i></b> | -         |

| Objectives                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Command                                                                | Reference |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|-----------|
| To apply a route-policy to public network IPv6 direct routes.<br><br><b>NOTE</b><br>Currently, QoS information, such as the traffic behavior or local ID, can be configured for direct routes only based on a route-policy. After QoS information is configured for routes, the device can apply different QoS policies to packets based on the QoS information during packet forwarding. In this manner, traffic statistic collection, authentication, and accounting can be performed based on direct routes. | <b>ipv6 direct-routing-table route-policy <i>route-policy-name</i></b> | -         |

c. Run **commit**

The configuration is committed.

- Apply a route-policy to static routes.

a. Run **system-view**

The system view is displayed.

b. Run **ip static-routing-table route-policy *route-policy-name***

A route-policy is applied to static routes on the public network.

c. Run **commit**

The configuration is committed.

- Apply a route-policy to IPv6 static routes.

a. Run **system-view**

The system view is displayed.

- b. Run **ipv6 static-routing-table route-policy route-policy-name**  
A route-policy is applied to public network static routes.
  - c. Run **commit**  
The configuration is committed.
- Apply a route-policy to RIP routes.
    - a. Run **system-view**  
The system view is displayed.
    - b. Run **rip [ process-id ]**  
A RIP process is enabled, and the RIP view is displayed.
    - c. Perform the following operations as required to apply a route-policy to RIP routes. For details, see [Table 1-168](#).

**Table 1-168** Applying a route-policy to RIP routes

| Objectives                                                                                                                                                                                                                                      | Command                                                                                                                                     | Reference                                                   |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|
| To configure a device to generate a default route or advertise a default route in the routing table to neighbors only when the matching conditions of a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) are met | <b>default-route originate</b><br>[ <b>cost cost</b>   <b>tag tag</b>   <b>route-policy route-policy-name</b> [ <b>avoid-learning</b> ] ] * | <a href="#">Configuring RIP to Advertise Default Routes</a> |
| To configure RIP to import the BGP routes that match a route-policy (specified by the <b>route-policy route-policy-name</b> parameter)                                                                                                          | <b>import-route bgp</b> [ <b>permit-ibgp</b> ] [ <b>cost { cost }</b>   <b>transparent</b> ]   <b>route-policy route-policy-name</b> ] *    | <a href="#">Configuring RIP to Import External Routes</a>   |

| Objectives                                                                                                                                                       | Command                                                                                                                               | Reference                                                 |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|
| To configure RIP to import the routes from another routing protocol that match a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) | <b>import-route { static   direct   unr   { isis   ospf   rip } [ process-id ] } [ cost cost   route-policy route-policy-name ] *</b> | <a href="#">Configuring RIP to Import External Routes</a> |
| To set a priority for RIP routes that match a route-policy (specified by the <b>route-policy route-policy-name</b> parameter)                                    | <b>preference { preference   route-policy route-policy-name } *</b>                                                                   | <a href="#">(Optional) Configuring a RIP Preference</a>   |

d. Run **commit**

The configuration is committed.

- Apply a route-policy to RIPng routes.
  - a. Run **system-view**
  - b. Run **ripng [ process-id ]**
  - c. Perform the following operations as required to apply a route-policy to RIPng routes. For details, see [Table 1-169](#).

**Table 1-169** Applying a route-policy to RIPng routes

| Objectives                                                                                                                                                         | Command                                                                                                                                                         | Reference                                                   |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|
| To configure RIPng to import the routes from another routing protocol that match a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) | <b>import-route { static   direct   bgp [ permit-ibgp ]   unr   { isis   ospfv3   ripng } [ process-id ] } [ cost cost   route-policy route-policy-name ] *</b> | <a href="#">Configuring RIPng to Import External Routes</a> |

| Objectives                                                                                                                             | Command                                                                                  | Reference                                            |
|----------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|------------------------------------------------------|
| To set a priority for RIPng routes that match a route-policy (specified by the <b>route-policy</b> <i>route-policy-name</i> parameter) | <b>preference { <i>preference</i>   <b>route-policy</b> <i>route-policy-name</i> } *</b> | <b>(Optional)<br/>Configuring the RIPng Priority</b> |

d. Run **commit**

The configuration is committed.

- Apply a route-policy to IPv4 IS-IS routes.
  - To apply a route-policy in the IS-IS view, perform the following operations:
    - i. Run the **system-view** command to enter the system view.
    - ii. Run the **isis [ process-id ]** command to enter the IS-IS view.
    - iii. Perform the following operations as required to apply a route-policy to IS-IS routes. For details, see [Table 1-170](#).

**Table 1-170** Applying a route-policy to IPv4 IS-IS routes in the IS-IS view

| Objectives                                                                                                                                                                                                                                                                                                                                                                                       | Command                                                                                                                                        | Reference                                                         |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| To configure IS-IS to generate and advertise default routes to the IS-IS domain only when external routes that match a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) exist in the routing table of a border device (This prevents routing blackhole when link faults make some important external routes unavailable but default routes are still advertised.) | <b>default-route-advertise route-policy route-policy-name [ cost cost   tag tag   [ level-1   level-1-2   level-2 ] ] * [ avoid-learning ]</b> | <a href="#">Configuring IS-IS to Generate IPv4 Default Routes</a> |
| To configure IS-IS to advertise the routes that are imported from another routing protocol and match a route-policy                                                                                                                                                                                                                                                                              | <b>filter-policy route-policy route-policy-name export [ protocol [ process-id ] ]</b>                                                         | -                                                                 |
| To configure IS-IS to accept the routes that match a route-policy                                                                                                                                                                                                                                                                                                                                | <b>filter-policy route-policy route-policy-name import</b>                                                                                     | <a href="#">Configuring IPv4 IS-IS to Import External Routes</a>  |

| Objectives                                                                                      | Command                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | Reference                                                        |
|-------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| To configure IS-IS to import the routes from another routing protocol that match a route-policy | <pre>import-route { direct   static   unr   { ospf   rip   isis } [ process-id ]   bgp [ permit-ibgp ] } [ cost-type { external   internal }   cost cost   tag tag   route-policy route-policy-name   [ level-1   level-2   level-1-2 ] ] *<br/> import-route { { ospf   rip   isis } [ process-id ]   bgp [ permit-ibgp ]   direct   unr } inherit-cost [ { level-1   level-2   level-1-2 }   tag tag   route-policy route-policy-name ] *<br/> import-route { ospf   isis } [ process-id ] [ cost-type { external   internal }   cost cost   tag tag   route-policy route-policy-name   [ level-1   level-2   level-1-2 ]   no-sid ] *<br/> import-route { ospf   isis } [ process-id ] inherit-cost [ { level-1   level-2   level-1-2 }   tag tag   route-policy route-policy-name   no-sid ] *</pre> | <a href="#">Configuring IPv4 IS-IS to Import External Routes</a> |
| To allow Level-1 routes to leak to a Level-2 area                                               | <code>import-route isis level-1 into level-2 [ filter-policy route-policy route-policy-name   direct allow-filter-policy   tag tag   no-sid ] *</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | <a href="#">Configuring IPv4 IS-IS Route Leaking</a>             |
| To allow Level-2 routes to leak to a Level-1 area                                               | <code>import-route isis level-2 into level-1 [ filter-policy route-policy route-policy-name   direct allow-filter-policy   tag tag   no-sid ] *</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | <a href="#">Configuring IPv4 IS-IS Route Leaking</a>             |

| Objectives                                                             | Command                                                             | Reference                                                     |
|------------------------------------------------------------------------|---------------------------------------------------------------------|---------------------------------------------------------------|
| To configure a priority for the IS-IS routes that match a route-policy | <b>preference { route-policy route-policy-name   preference } *</b> | <a href="#">Configuring a Preference Value for IPv4 IS-IS</a> |

- iv. Run the **commit** command to commit the configuration.
- To apply a route-policy in the IS-IS FRR view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **isis [ process-id ]** command to enter the IS-IS view.
  - iii. Run the **frr** command to enter the IS-IS FRR view.
  - iv. Run the **frr-policy route route-policy route-policy-name** command to configure IS-IS to add the backup routes that match a route-policy to the IP routing table.  
For details on how to configure IS-IS Auto FRR (IPv4), see [1.1.8.2.18 Configuring IS-IS Auto FRR](#).
  - v. Run the **commit** command to commit the configuration.
- Apply a route-policy to IPv6 IS-IS routes.
  - To apply a route-policy in the IS-IS view, perform the following operations:
    - i. Run the **system-view** command to enter the system view.
    - ii. Run the **isis [ process-id ]** command to enter the IS-IS view.
    - iii. Perform the following operations as required to apply a route-policy to IPv6 IS-IS routes. For details, see [Table 1-171](#).

**Table 1-171** Applying a route-policy to IPv6 IS-IS routes in the IS-IS view

| Objectives                                                                                                                                                                                                                                                | Command                                                                                                                                                     | Reference                                                         |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|
| To configure IS-IS to generate and advertise default IPv6 routes to the IS-IS domain only when external routes that match a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) exist in the routing table of a border device | <b>ipv6 default-route-advertise route-policy-name [ cost cost   tag tag   [ level-1   level-2   level-1-2 ] ] * [ avoid-learning  learning-avoid-loop ]</b> | <a href="#">Configuring IS-IS to Generate IPv6 Default Routes</a> |
| To configure IS-IS to advertise the routes that are imported from another routing protocol and match a route-policy                                                                                                                                       | <b>ipv6 filter-policy route-policy route-policy-name export [ protocol [ process-id ] ]</b>                                                                 | -                                                                 |
| To configure IS-IS to accept the IPv6 routes that match a route-policy                                                                                                                                                                                    | <b>ipv6 filter-policy route-policy route-policy-name import</b>                                                                                             | <a href="#">Configuring IPv6 IS-IS to Import External Routes</a>  |

| Objectives                                                                                      | Command                                                                                                                                                                                                                                                                                                                                                                                                                                      | Reference                                                        |
|-------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| To configure IS-IS to import the routes from another routing protocol that match a route-policy | <pre>ipv6 import-route { direct   static   unr   { ripng   isis   ospfv3 } [ process-id ] } [ bgp [ permit-ibgp ] } [ cost cost   tag tag   route- policy route-policy-name   [ level-1   level-2   level-1-2 ] ] *</pre><br><pre>ipv6 import-route { { ospfv3   ripng   isis } [ process-id ]   bgp [ permit-ibgp ]   direct   unr } inherit-cost [ tag tag   route-policy route- policy-name   [ level-1   level-2   level-1-2 ] ] *</pre> | <a href="#">Configuring IPv6 IS-IS to Import External Routes</a> |
| To configure Level-1 IPv6 routes that match a route-policy to leak to a Level-2 area            | <b>ipv6 import-route isis level-1 into level-2 [ filter-policy route- policy route-policy-name   direct allow-filter-policy   tag tag ] *</b>                                                                                                                                                                                                                                                                                                | <a href="#">Configuring IPv6 IS-IS Route Leaking</a>             |
| To configure Level-2 IPv6 routes that match a route-policy to leak to a Level-1 area            | <b>ipv6 import-route isis level-2 into level-1 [ filter-policy route- policy route-policy-name   direct allow-filter-policy   tag tag ] *</b>                                                                                                                                                                                                                                                                                                | <a href="#">Configuring IPv6 IS-IS Route Leaking</a>             |
| To configure a priority for the IS-IS routes that match a route-policy                          | <b>ipv6 preference { route- policy route-policy-name   preference } *</b>                                                                                                                                                                                                                                                                                                                                                                    | <a href="#">Configuring a Preference Value for IPv6 IS-IS</a>    |

- iv. Run the **commit** command to commit the configuration.
- To apply a route-policy in the IS-IS IPv6 topology view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **isis [ process-id ]** command to enter the IS-IS view.
  - iii. Run the **ipv6 topology topology-name [ topology-id { multicast |
topology-id } ]** command to bind the IS-IS process to an IPv6 topology and enter the IS-IS IPv6 topology view.

- iv. Run the **import-route { direct | isis [ process-id ] inherit-cost [ tag tag | route-policy route-policy-name | { level-1 | level-2 | level-1-2 } ] }** \* command to configure IS-IS to import routes from another protocol into the current topology.  
For details on how to configure IPv6 IS-IS multi-topology, see [Enabling MT for an IS-IS Process](#).
- v. Run the **commit** command to commit the configuration.
- Apply a route-policy to OSPF routes.
  - To apply a route-policy in the OSPF view, perform the following operations:
    - i. Run the **system-view** command to enter the system view.
    - ii. Run the **ospf [ process-id ]** command to enable an OSPF process and enter the OSPF view.
    - iii. To apply a route-policy to OSPF routes in the OSPF process view, see [Table 1-172](#).

**Table 1-172** Applying a route-policy to OSPF routes

| Objectives                                                                                                                                                                                                                          | Command                                                                                                                                                                                                                                                                                                                                                                                 | Reference                                                              |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|
| To configure OSPF to advertise the default routes in the routing table that are not generated by OSPF to a common area based on the parameters of a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) | <b>default-route-advertise</b> [ [ always   permit-calculate-other ]   cost cost   type type   route-policy route-policy-name   distribute-delay delay-time ] *<br><b>default-route-advertise</b> [ permit-calculate-other   cost cost   type type   route-policy route-policy-name   distribute-delay delay-time   permit-ibgp ] *<br><b>default-route-advertise summary cost cost</b> | <a href="#">Configuring OSPF to Import a Default Route</a>             |
| To configure OSPF to accept the routes that match a route-policy                                                                                                                                                                    | <b>filter-policy route-policy route-policy-name [ secondary ] import</b>                                                                                                                                                                                                                                                                                                                | <a href="#">Configuring OSPF to Filter Received Routes</a>             |
| To configure OSPF to import the routes that match a route-policy                                                                                                                                                                    | <b>filter-policy route-policy route-policy-name export [ protocol [ process-id ] ]</b>                                                                                                                                                                                                                                                                                                  | <a href="#">Configuring OSPF to Filter the Routes to Be Advertised</a> |

| Objectives                                                                                                                                                                                               | Command                                                                                                                                                                                                                   | Reference                                                  |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| To configure OSPF to import routes from another protocol                                                                                                                                                 | <b>import-route { bgp [ permit-ibgp ]   direct   unr   rip [ process-id-rip ]   static   isis [ process-id-isis ]   ospf [ process-id-ospf ] [ cost cost   route-policy route-policy-name   tag tag   type type ] * }</b> | <a href="#">Configuring OSPF to Import External Routes</a> |
| To configure a route-policy for OSPF local MT so that only the routes that match the route-policy (specified by the <b>route-policy route-policy-name</b> parameter) are added to the MIGP routing table | <b>local-mt filter-policy route-policy route-policy-name</b>                                                                                                                                                              | -                                                          |
| To configure a priority for OSPF routes that match a route-policy                                                                                                                                        | <b>preference [ ase   inter   intra ] { preference   route-policy route-policy-name } *</b>                                                                                                                               | <a href="#">(Optional) Setting the OSPF Preference</a>     |

- iv. Run the **commit** command to commit the configuration.
- To apply a route-policy in the OSPF area view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **ospf [ process-id ]** command to enable an OSPF process and enter the OSPF view.
  - iii. Run the **area area-id** command to enter the OSPF area view.
  - iv. Perform either of the following operations to apply a route-policy in the OSPF area view:
    - o Run the **filter route-policy route-policy-name export** command to apply a route-policy to outgoing Type 3 LSAs (summary LSAs) in the area.
    - o Run the **filter route-policy route-policy-name import** command to apply a route-policy to incoming Type 3 LSAs in the area.

For details on the configuration of applying a route-policy to Type-3 LSAs, see [Configuring OSPF to Filter LSAs in an Area](#)

- v. Run the **commit** command to commit the configuration.
  - To apply a route-policy in the OSPF FRR view, perform the following operations:
    - i. Run the **system-view** command to enter the system view.
    - ii. Run the **ospf [ process-id ]** command to enable an OSPF process and enter the OSPF view.
    - iii. Run the **frr** command to enter the OSPF FRR view.
    - iv. Run the **loop-free-alternate** command to enable OSPF IP FRR to generate a loop-free backup link.
    - v. Run the **frr-policy route route-policy route-policy-name** command to configure OSPF to add the backup routes that match a route-policy to the IP routing table.
  - For details on how to configure OSPF IP FRR, see [1.1.4.2.19 Configuring OSPF IP FRR](#).
  - vi. Run the **commit** command to commit the configuration.
- Apply a route-policy to OSPFv3 routes.
    - To apply a route-policy in the OSPFv3 view, perform the following operations:
      - i. Run the **system-view** command to enter the system view.
      - ii. Run the **ospfv3 [ process-id ]** command to enable an OSPFv3 process and enter the OSPFv3 view.
      - iii. Perform the following operations as required to apply a route-policy in the OSPFv3 view. For details, see [Table 1-173](#).

**Table 1-173** Applying a route-policy to OSPFv3 routes

| Objectives                                                                                                                                                                                                                                       | Command                                                                                                                                                                                                                                                       | Reference                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| To configure OSPFv3 to advertise the default routes in the routing table that are not generated by OSPFv3 to an OSPFv3 routing area based on the parameters of a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) | <b>default-route-advertise</b> [ <b>always</b>   <b>permit</b>   <b>calculate-other</b>   <b>cost</b> <b>cost</b>   <b>type</b> <b>type</b>   <b>tag</b> <b>tag</b>   <b>distribute-delay</b> <b>delay</b>   <b>route-policy</b> <b>route-policy-name</b> ] * | <a href="#">Configuring OSPFv3 to Filter the Routes to Be Advertised</a> |

| Objectives                                                         | Command                                                                                                                                                                                                                                                                                                                                                | Reference                                                                |
|--------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|
| To configure OSPFv3 to accept the routes that match a route-policy | <b>filter-policy route-policy</b> <i>route-policy-name</i> [ <b>secondary</b> ] <b>import</b>                                                                                                                                                                                                                                                          | <a href="#">Configuring OSPFv3 to Filter Received Routes</a>             |
| To configure OSPFv3 to import the routes that match a route-policy | <b>filter-policy route-policy</b> <i>route-policy-name</i> <b>export</b> [ <i>protocol</i> [ <i>process-id</i> ]]                                                                                                                                                                                                                                      | <a href="#">Configuring OSPFv3 to Filter the Routes to Be Advertised</a> |
| To configure OSPFv3 to import routes from another protocol         | <b>import-route</b> { <b>bgp</b> [ <b>permit-ibgp</b> ]   <b>direct</b>   <b>unr</b>   <b>static</b>   <b>isis</b> [ <i>process-id</i> ]   <b>ripng</b> [ <i>process-id</i> ]   <b>ospfv3</b> [ <i>process-id</i> ] } [ <b>cost</b> <i>cost</i>   <b>tag</b> <i>tag</i>   <b>type</b> <i>type</i>   <b>route-policy</b> <i>route-policy-name</i> ] * } | <a href="#">Configuring OSPFv3 to Import External Routes</a>             |
| To configure a priority for OSPF routes that match a route-policy  | <b>preference</b> [ <i>ase</i> ] { <i>preference</i>   <b>route-policy</b> <i>route-policy-name</i> } *                                                                                                                                                                                                                                                | -                                                                        |

- iv. Run the **commit** command to commit the configuration.
- To apply a route-policy in the OSPFv3 area view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **ospfv3** [*process-id*] command to enable an OSPFv3 process and enter the OSPFv3 view.
  - iii. Run the **area** *area-id* command to enter the OSPFv3 area view.
  - iv. Perform either of the following operations to apply a route-policy in the OSPFv3 area view:
    - o Run the **filter route-policy** *route-policy-name* **export** command to apply a route-policy to outgoing Type 3 LSAs (Inter-Area-Prefix-LSAs) in the OSPFv3 area.
    - o Run the **filter route-policy** *route-policy-name* **import** command to apply a route-policy to incoming Type 3 LSAs in the OSPFv3 area.

For details on how to apply a route-policy to Type 3 LSAs in an OSPFv3 area, see [Configuring OSPFv3 to Filter LSAs in an Area](#).

- v. Run the **commit** command to commit the configuration.

- To apply a route-policy in the OSPFv3 FRR view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **ospfv3 [ process-id ]** command to enable an OSPFv3 process and enter the OSPFv3 view.
  - iii. Run the **frr** command to enter the OSPFv3 IP FRR view.
  - iv. Run the **loop-free-alternate** command to enable OSPFv3 IP FRR.
  - v. Run the **frr-policy route route-policy route-policy-name** command to configure OSPFv3 to add the backup routes that match a route-policy to the IP routing table.
- For details on how to configure OSPFv3 IP FRR, see [1.1.5.2.16 Configuring OSPFv3 IP FRR](#).
- vi. Run the **commit** command to commit the configuration.
- Apply a route-policy to BGP routes.
  - a. Run the **system-view** command to enter the system view.
  - b. Run the **bgp { as-number-plain | as-number-dot }** command to enter the BGP view.
  - c. Run the **ipv4-family unicast** command to enter the IPv4 unicast address family view.
  - d. Perform the following operations as required to apply a route-policy to BGP routes. For details, see [Table 1-174](#).

**Table 1-174** Applying a route-policy to BGP routes

| Objectives                                                      | Command                                                                                                                                                                                             | Reference                                                     |
|-----------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------|
| To configure a summary BGP route                                | <b>aggregate ipv4-address { mask   mask-length } [ as-set   attribute-policy route-policy-name1   detail-suppressed   origin-policy route-policy-name2   suppress-policy route-policy-name3 ] *</b> | <a href="#">1.1.9.2.8 Configuring BGP Route Summarization</a> |
| To configure BGP route dampening                                | <b>dampening [ half-life-reach reuse suppress ceiling   route-policy route-policy-name ] *[ update-standard ]</b>                                                                                   | <a href="#">1.1.9.2.45 Configuring BGP Route Dampening</a>    |
| To configure BGP to import routes from another routing protocol | <b>import-route protocol [ process-id ] [ med med   route-policy route-policy-name ] *[ non-relay-tunnel ]</b>                                                                                      | <a href="#">Configuring BGP to Import Routes</a>              |

| Objectives                                                                                                                                                                                                                                                                                                                                  | Command                                                                                                                                                                                                                                                                                                                                                           | Reference                                                                               |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
| To configure BGP to import local routes                                                                                                                                                                                                                                                                                                     | <b>network</b> <i>ipv4-address</i> [ <i>mask</i>   <i>mask-length</i> ] [ <b>route-policy</b> <i>route-policy-name</i> ] [ <b>non-relay-tunnel</b> ]                                                                                                                                                                                                              | <a href="#">Configuring BGP to Import Routes</a>                                        |
| To enable BGP route recursion based on a route-policy                                                                                                                                                                                                                                                                                       | <b>nexthop recursive-lookup</b><br><b>route-policy</b> <i>route-policy-name</i>                                                                                                                                                                                                                                                                                   | <a href="#">Configuring the Next_Hop Attribute</a>                                      |
| To configure a BGP device to send a default route to a peer or a peer group and use a route-policy (specified by the <b>route-policy</b> <i>route-policy-name</i> parameter) to modify the attributes of the default route                                                                                                                  | <b>peer</b> { <i>group-name</i>   <i>ipv4-address</i> } <b>default-route-advertise</b> [ <b>route-policy</b> <i>route-policy-name</i> ] [ <b>conditional-route-match-all</b> { <i>ipv4-address1</i> { <i>mask1</i>   <i>mask-length1</i> } } &<1-4>   <b>conditional-route-match-any</b> { <i>ipv4-address2</i> { <i>mask2</i>   <i>mask-length2</i> } } &<1-4> ] | <a href="#">1.1.9.2.20 Configuring a BGP Device to Send a Default Route to Its Peer</a> |
| To configure BGP to ignore bit error detection results when an export route-policy is used.<br><br><b>NOTE</b><br>When bit-error-triggered protection switching is configured, bit errors occur, and Local_Pref or MED is modified using an export route-policy, run the command to apply the Local_Pref or MED in the export route-policy. | <b>peer</b> { <i>group-name</i>   <i>ipv4-address</i> } <b>route-policy</b> <i>route-policy-name</i> <b>export</b> <b>ignore-bit-error</b>                                                                                                                                                                                                                        | -                                                                                       |

| Objectives                                                                                                                                                               | Command                                                                                      | Reference                                                                                                                 |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| To configure a device to accept the routes that match a route-policy from a peer or peer group or advertise the routes that match a route-policy to a peer or peer group | <b>peer { group-name   ipv4-address } route-policy route-policy-name { import   export }</b> | <a href="#">Applying a Policy to BGP Route Advertisement</a><br><a href="#">Applying a Policy to BGP Route Acceptance</a> |
| To use a route-policy to set a BGP priority                                                                                                                              | <b>preference route-policy route-policy-name</b>                                             | <a href="#">Setting the BGP Preference</a>                                                                                |
| To prevent the BGP routes that match a route-policy from being added to the IP routing table                                                                             | <b>routing-table rib-only [ route-policy route-policy-name ]</b>                             | <a href="#">(Optional) Preventing a Device from Adding BGP Routes to the IP Routing Table</a>                             |

- e. Run the **commit** command to commit the configuration.
- Apply a route-policy to BGP4+ routes.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **bgp { as-number-plain | as-number-dot }**

The BGP view is displayed.
  - c. Run **ipv6-family [ unicast ]**

The IPv6 unicast address family view is displayed.
  - d. Perform the following operations as required to apply a route-policy to BGP4+ routes. For details, see [Table 1-175](#).

**Table 1-175 Applying a route-policy to BGP4+ routes**

| Objectives                                                                                                                                                                                                                                                                            | Command                                                                                                                                                                                                                                                   | Reference                                                     |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------|
| To configure a summary BGP4+ route                                                                                                                                                                                                                                                    | <b>aggregate</b> <i>ipv6-address prefix-length</i> [ <b>as-set</b>   <b>attribute-policy</b> <i>route-policy-name1</i>   <b>detail-suppressed</b>   <b>origin-policy</b> <i>route-policy-name2</i>   <b>suppress-policy</b> <i>route-policy-name3</i> ] * | <a href="#">Configuring BGP4+ Route Summarization</a>         |
| To configure BGP4+ route dampening and apply dampening parameters to the routes that match a route-policy (specified by the <b>route-policy</b> <i>route-policy-name</i> parameter) so that specified requirement dampening can be performed by BGP based on the applied route-policy | <b>dampening</b> [ <i>half-life-reach reuse suppress ceiling</i>   <b>route-policy</b> <i>route-policy-name</i> ] *                                                                                                                                       | <a href="#">1.1.10.1.25 Configuring BGP4+ Route Dampening</a> |
| To configure BGP4+ to import the routes from another routing protocol that match a route-policy                                                                                                                                                                                       | <b>import-route</b> <i>protocol</i> [ <i>process-id</i> ] [ <b>med</b> <i>med</i>   <b>route-policy</b> <i>route-policy-name</i> ] *                                                                                                                      | <a href="#">Configuring BGP4+ to Import Routes</a>            |
| To configure BGP4+ to advertise local routes                                                                                                                                                                                                                                          | <b>network</b> <i>ipv6-address prefix-length</i> [ <b>route-policy</b> <i>route-policy-name</i> ]                                                                                                                                                         | <a href="#">Configuring BGP4+ to Import Routes</a>            |
| To enable BGP4+ route recursion based on a route-policy                                                                                                                                                                                                                               | <b>nexthop recursive-lookup</b> <b>route-policy</b> <i>route-policy-name</i>                                                                                                                                                                              | <a href="#">Setting the Next_Hop Attribute</a>                |

| Objectives                                                                                                                                                                                                            | Command                                                                                                                                                                                                            | Reference                                                                                                                         |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| To configure a BGP4+ device to send a default route to a peer or a peer group and use a route-policy (specified by the <b>route-policy route-policy-name</b> parameter) to modify the attributes of the default route | <b>peer { group-name   ipv6-address } default-route-advertise [ route-policy route-policy-name ] { conditional-route-match-all   conditional-route-match-any } { ipv6-address mask-length } } &amp;&lt;1-4&gt;</b> | <a href="#">Configuring BGP4+ to Advertise Default Routes to Peers or peer groups</a>                                             |
| To configure a device to accept the routes that match a route-policy from a peer or peer group or advertise the routes that match a route-policy to a peer or peer group                                              | <b>peer { group-name   ipv6-address } route-policy route-policy-name { import   export }</b>                                                                                                                       | <a href="#">Applying a Policy to BGP4+ Route Advertisement</a><br><a href="#">Configuring a Policy for Receiving BGP4+ Routes</a> |
| To use a route-policy to set a BGP4+ priority                                                                                                                                                                         | <b>preference route-policy route-policy-name</b>                                                                                                                                                                   | <a href="#">Setting a BGP4+ Preference</a>                                                                                        |
| To prevent the BGP4+ routes that match a route-policy from being added to the IPv6 routing table                                                                                                                      | <b>routing-table rib-only [ route-policy route-policy-name ]</b>                                                                                                                                                   | <a href="#">(Optional) Preventing BGP4+ from Adding Routes to the IP Routing Table</a>                                            |

e. Run **commit**

The configuration is committed.

- Apply a route-policy to BGP/MPLS IP VPN routes.
  - To apply a route-policy in the BGP-VPNv4 address family view, perform the following operations:
    - i. Run the **system-view** command to enter the system view.
    - ii. Run the **bgp { as-number-plain | as-number-dot }** command to enter the BGP view.

- iii. Run the **ipv4-family vpnv4** command to enter the BGP-VPNV4 address family view.
  - iv. Run the **nexthop recursive-lookup bit-error-detection { med + med-adjust-value | local-preference - localpref-adjust-value } \* [ route-policy route-policy-name ]** command to associate bit error events with the adjustment of the MED or Local\_Pref value for routes that match a route-policy. If **route-policy route-policy-name** is not specified in the command, the MED or Local\_Pref values of all routes are adjusted.  
For details on how to apply a route-policy to bit-error-triggered L3VPN route switching, see Configuring Bit-Error-Triggered L3VPN Route Switching.
  - v. Run the **commit** command to commit the configuration.
- To apply a route-policy in the VPN instance view, perform the following operations:
- i. Run the **system-view** command to enter the system view.
  - ii. Run the **ip vpn-instance vpn-instance-name** command to enter the VPN instance view.
  - iii. To apply a route-policy to BGP/MPLS IP VPN routes in the VPN instance view, perform one of the following operations as required:
    - o To associate the current address family of the VPN instance with an export route-policy, run the **export route-policy route-policy-name [ add-ert-first ]** command. Only one route-policy can be associated with the current address family. If the command is run more than once and the specified route-policies are different, the route-policy specified in the latest configuration overrides the previous ones.  
The **export route-policy** command can control route advertisement between different VPN instances on the same PE, whereas the **peer route-policy export** command can control only the VPNv4 or VPNv6 routes that a PE sends to other PE peers.
    - o To associate the current address family of the VPN instance with an import route-policy, run the **import route-policy route-policy-name** command. Only one route-policy can be associated with the current address family. If the command is run more than once and the specified route-policies are different, the route-policy specified in the latest configuration overrides the previous ones.  
The **import route-policy** command can control route acceptance between different VPN instances on the same PE, whereas the **peer route-policy import** command can control only the VPNv4 or VPNv6 routes that a PE accepts from other PE peers.
    - o To configure the device to advertise VPN ARP Vlink direct routes, run the **arp vlink-direct-route advertise [ route-policy route-policy-name ]** command. The **route-policy route-policy-name** parameter specifies the route-policy used to filter the ARP Vlink direct routes to be advertised. For details, see Configuring Route Exchange Between PEs and CEs.

- iv. Run the **commit** command to commit the configuration.
- To apply a route-policy in the VPN instance IPv4 address family view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **ip vpn-instance *vpn-instance-name*** command to enter the VPN instance view.
  - iii. Run the **ipv4-family** command to enable the IPv4 address family for the VPN instance and enter the VPN instance IPv4 address family view.
  - iv. Run the **ip { direct-routing-table | static-routing-table } route-policy *route-policy-name*** command to apply a route-policy to the direct or static routes in the VPN instance IPv4 address family.  
If the **ip route-policy** command is run, the device modifies a specified attribute of the direct or static routes that match the specified route-policy.
  - v. Run the **commit** command to commit the configuration.
- To apply a route-policy in the VPN instance IPv6 address family view, perform the following operations:
  - i. Run the **system-view** command to enter the system view.
  - ii. Run the **ip vpn-instance *vpn-instance-name*** command to enter the VPN instance view.
  - iii. Run the **ipv6-family** command to enable the IPv6 address family for the VPN instance and enter the VPN instance IPv6 address family view.
  - iv. Run the **ipv6 { direct-routing-table | static-routing-table } route-policy *route-policy-name*** command to apply a route-policy to the direct or static routes in the VPN instance IPv6 address family.
  - v. Run the **nd vlink-direct-route advertise [ route-policy *route-policy-name* ]** command to configure the device to advertise VPN NDP Vlink direct routes. The **route-policy *route-policy-name*** parameter specifies the route-policy used to filter the NDP Vlink direct routes to be advertised. For details, see Configuring Route Exchange Between PEs and CEs.
  - vi. Run the **commit** command to commit the configuration.

----End

## Verifying the Route-Policy Configuration

After configuring a route-policy, verify information about the route-policy.

### Prerequisites

The route-policy has been configured.

### Procedure

**Step 1** Run the **display route-policy [ *route-policy-name* ]** command to check information about the route-policy.

**Step 2** (Optional) Run the **reset route-policy route-policy-name counters** command to reset route-policy counters.

----End

### 1.1.11.2.11 Applying Filters to Received Routes

By applying filters of routing policies to routing protocols, you can filter received routes.

#### Usage Scenario

When exchanging routes on a network, devices need to accept only required routes. After defining a filter (such as the IP prefix list, ACL, or route-policy) of a routing policy, you need to apply the filter to routing protocols. You can use the **filter-policy** command in the protocol view and apply an ACL or an IP prefix list to filter the received routes. Only the routes that meet the matching rules are accepted.

The **filter-policy import** command is used to filter received routes. The influence of running the **filter-policy** command for a distance-vector protocol is different from that for a link-state protocol.

- Distance-vector protocol

Distance-vector protocols generate routes based on their routing tables. Running the preceding command filters the routes received from neighbors and the routes to be advertised to neighbors.

- Link-state protocol

Link-state routing protocols generate routes based on their LSDBs. Running the **filter-policy** command does not affect the integrity of link state advertisements or LSDBs. Therefore, running the **filter-policy** command has different impacts on route acceptance and advertisement.

After routes are received, the **filter-policy** command determines which routes are to be added from the protocol routing table to the local core routing table. Therefore, running this command affects the local core routing table rather than the protocol routing table.

#### NOTE

- BGP has powerful filtering functions. For the configuration of BGP routing policies, see "BGP Configuration."
- For details about **filter-policy** and **import-route** commands and their applications in RIP, OSPF, IS-IS, and BGP, see related configurations.

#### Pre-configuration Tasks

Before applying filters to imported routes, complete the following tasks:

- [Configure an IP prefix list](#).
- Configure an ACL.
- [Configure a route-policy](#).

#### Configuring RIP to Filter the Received Routes

You can configure an inbound or outbound filtering policy by specifying Access Control Lists (ACLs) and IP address prefix lists to filter routes to be received and

advertised. You can also configure a device to receive only the RIP packets from a specified neighbor.

## Context

Devices can filter the routing information. To filter the received and advertised routes, you can configure inbound and outbound filtering policies by specifying the ACL and IP prefix list.

You can also configure a device to receive RIP packets from only a specified neighbor.

## Procedure

### Step 1 Run **system-view**

The system view is displayed.

### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

### Step 3 Set the conditions to filter the received routes.

Run any of the following commands as required:

- Based on the basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The basic ACL view is displayed.

- c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule for the basic ACL is configured.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.

- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.

- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- d. Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

- e. Run **filter-policy { acl-number | acl-name acl-name } import [ interface-type interface-number ]**

An import policy that is based on the basic ACL is configured to filter routes that are received in RIP update packets.

- Based on the IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name import [ interface-type interface-number ]**

An import policy that is based on the IP prefix list is configured to filter routes that are received in RIP update packets.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring OSPF to Filter the Received Routes

After a filtering policy is configured for the OSPF routes that need to be delivered to the routing management module, only the routes that match the policy will be added to the routing table.

## Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run **ospf [ process-id ]**

The OSPF view is displayed.

**Step 3** Set the conditions to filter the received routes.

Run any of the following commands as required:

- Based on the basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The basic ACL view is displayed.

- c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule for the basic ACL is configured.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- d. Run **ospf [ process-id ]**

The OSPF view is displayed.

e. Run **filter-policy { acl-number | acl-name acl-name } import**

An import policy that is based on the basic ACL is configured to filter routes received by OSPF.

- Based on the IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name import**

An import policy that is based on the IP prefix list is configured to filter routes received by OSPF.

**Step 4** Run **commit**

The configuration is committed.

----End

## Configuring IS-IS to Filter the Received Routes

By configuring IS-IS to filter the received routes, you can control the number of IS-IS routes to be added to the IP routing table, and thus reduce the size of the IP routing table.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

The IS-IS view is displayed.

**Step 3** Set the conditions to filter the received routes.

Run any of the following commands as required:

- Based on the basic ACL:

a. Run **quit**

Return to the system view.

b. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The basic ACL view is displayed.

c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule for the basic ACL is configured.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:
  - Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.
  - Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

d. Run **isis [ process-id ]**

The IS-IS view is displayed.

e. Run **filter-policy { acl-number | acl-name acl-name } import**

An import policy that is based on the basic ACL is configured to filter routes received by IS-IS.

• Based on the IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name import**

An import policy that is based on the IP prefix list is configured to filter routes received by IS-IS.

#### Step 4 Run **commit**

The configuration is committed.

----End

### Verifying the Configuration

After applying filters to the received routes, check information about the routing table of each protocol.

### Prerequisites

Filters have been applied to the received routes.

## Procedure

- Run the **display rip process-id route** command to check information about the RIP routing table.
- Run the **display ospf [ process-id ] routing** command to check information about the OSPF routing table.
- Run the **display isis [ process-id ] route** command to check information about the IS-IS routing table.
- Run the **display ip routing-table** command to check information about the IP routing table.

Run the **display ip routing-table** command on the local router, and you can view that the routes matching the filtering rules of the neighbor have been filtered out or the **apply** action has been performed.

----End

### 1.1.11.2.12 Applying Filters to Routes to Be Advertised

By applying filters of routing policies to routing protocols, you can filter routes to be advertised.

## Usage Scenario

To enable a device to advertise required routes, define the filter (such as the IP prefix list, ACL, or route-policy) for a routing policy, apply the filter to routing protocols, and run the **filter-policy** command specified with the filter in the related protocol view to filter the routes to be advertised.

The function of the **filter-policy export** command varies the protocol type. And the functions to a distance-vector protocol and a link-state protocol are as follows:

- Distance-vector protocol  
A distance-vector protocol generates routes based on the routing table. Therefore, the command filters the routes received from neighbors and the routes to be advertised to neighbors.
- Link-state protocol  
A link-state protocol generates routes based on the LSDB. The **filter-policy** command does not affect any Link State Advertisement (LSA) or LSDB.  
When advertising routes, you can run the **filter-policy export** command to determine whether to advertise the imported routes (such as the imported RIP routes). Only the LSAs or Link State PDUs (LSPs) that are imported using the **filter-policy import** command are added to the LSDB. This does not affect the LSAs advertised to other routers.

#### NOTE

- BGP has the powerful filtering function. For the configuration of BGP routing policies, refer to "BGP Configuration."
- For details of the **filter-policy** and **import-route** commands and their applications in RIP, OSPF, IS-IS, and BGP, refer to related configurations.

## Pre-configuration Tasks

Before applying filters to routes to be advertised, complete the following tasks:

- [Configure an IP prefix list.](#)
- Configure an ACL.
- [Configure a route-policy.](#)

## Configuring RIP to Filter the Routes to Be Advertised

You can set conditions to filter the routes to be advertised. Only the routes that meet the conditions can be advertised.

### Context

Devices can filter the routing information. To filter the routes to be advertised, you can configure an export filtering policy by specifying the ACL and IP prefix list.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

#### Step 3 Set the conditions to filter routes to be advertised.

Run any of the following commands as required:

- Based on the basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The basic ACL view is displayed.

- c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule for the basic ACL is configured.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.

- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:
  - Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.
  - Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

d. Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

e. Run **filter-policy { acl-number | acl-name acl-name } export [ protocol process-id | interface-type interface-number ]**

An export policy that is based on the basic ACL is configured for RIP to filter the imported routes to be advertised.

• Based on the IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name export [ protocol process-id | interface-type interface-number ]**

An export policy that is based on the IP prefix list is configured for RIP to filter the imported routes to be advertised.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring OSPF to Filter the Routes to Be Advertised

After a filtering policy is configured for routes imported by OSPF, only the routes that match the policy can be advertised.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run **ospf [ process-id ]**

The OSPF view is displayed.

**Step 3** Set the conditions to filter routes to be advertised.

Run any of the following commands as required:

- Based on a basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The basic ACL view is displayed.

- c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ]**

A rule is configured for the ACL.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- d. Run **ospf [ process-id ]**

The OSPF view is displayed.

- e. Run **filter-policy { acl-number | acl-name acl-name } export [ direct | static | unr | bgp | { rip | isis | ospf } [ process-id ] ]**

The device is configured to filter the imported routes to be advertised.

- Based on an IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name export [ direct | static | unr | bgp | { rip | isis | ospf } [ process-id ] ]**

The device is configured to filter the imported routes to be advertised.

- Based on a route-policy:

Run **filter-policy route-policy route-policy-name export [ direct | static | unr | bgp | { rip | isis | ospf } [ process-id ] ]**

The device is configured to filter the imported routes to be advertised.

#### Step 4 Run commit

The configuration is committed.

----End

### Configuring IS-IS to Filter the Routes to Be Advertised

By configuring IS-IS to filter the routes to be advertised, you can effectively control the number of IS-IS routes on the network.

### Procedure

#### Step 1 Run system-view

The system view is displayed.

#### Step 2 Run isis [ process-id ]

The IS-IS view is displayed.

#### Step 3 Set the conditions to filter routes to be advertised.

Run any of the following commands as required:

- Based on the basic ACL:

- a. Run **quit**

Return to the system view.

- b. Run **acl { name basic-acl-name { basic | [ basic ] number basic-acl-number } | [ number ] basic-acl-number } [ match-order { config | auto } ]**

The basic ACL view is displayed.

- c. Run **rule [ rule-id ] [ name rule-name ] { deny | permit } [ fragment-type { fragment | non-fragment | non-subseq | fragment-subseq | fragment-spe-first } | source { source-ip-address { source-wildcard | 0 | src-netmask } | any } | time-range time-name | vpn-instance vpn-instance-name ] \***

A rule for the basic ACL is configured.

When the **rule** command is run to configure rules for a named ACL, only the source address range specified by **source** and the time period specified by **time-range** are valid as the rules.

When a filtering policy of a routing protocol is used to filter routes:

- If the action specified in an ACL rule is **permit**, a route that matches the rule will be received or advertised by the system.
- If the action specified in an ACL rule is **deny**, a route that matches the rule will not be received or advertised by the system.
- If a route has not matched any ACL rules, the route will not be received or advertised by the system.
- If an ACL does not contain any rules, all routes matching the **route-policy** that references the ACL will not be received or advertised by the system.
- In the configuration order, the system first matches a route with a rule that has a smaller number and then matches the route with a rule with a larger number. Routes can be filtered using a blacklist or a whitelist:

Route filtering using a blacklist: Configure a rule with a smaller number and specify the action **deny** in this rule to filter out the unwanted routes. Then, configure another rule with a larger number in the same ACL and specify the action **permit** in this rule to receive or advertise the other routes.

Route filtering using a whitelist: Configure a rule with a smaller number and specify the action **permit** in this rule to permit the routes to be received or advertised by the system. Then, configure another rule with a larger number in the same ACL and specify the action **deny** in this rule to filter out unwanted routes.

- d. Run **isis [ process-id ]**

The IS-IS view is displayed.

- e. Run **filter-policy { acl-number | acl-name acl-name } export [ direct | static | rip process-id | bgp | ospf process-id | isis process-id | unr ]**

An export policy that is based on the basic ACL is configured for IS-IS to filter the imported routes to be advertised.

- Based on the IP prefix list:

Run **filter-policy ip-prefix ip-prefix-name export [ direct | static | rip process-id | bgp | ospf process-id | isis process-id | unr ]**

An export policy that is based on the IP prefix list is configured for IS-IS to filter the imported routes to be advertised.

#### Step 4 Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After applying filters to the routes to be advertised, check information about the routing table of each protocol.

### Prerequisites

Filters have been applied to the routes to be advertised.

### Procedure

- Run the **display rip [ process-id ] route** command to check information about the RIP routing table.
- Run the **display ospf [ process-id ] routing** command to check information about the OSPF routing table.
- Run the **display isis [ process-id ] route** command to check information about the IS-IS routing table.
- Run the **display ip routing-table** command to check information about the IP routing table.

Run the **display ip routing-table** command on the neighbor, and you can view that the routes matching the filtering rules of the neighbor have been filtered out or the **apply** action has been performed.

----End

### 1.1.11.2.13 Applying Filters to Imported Routes

By applying filters of routing policies to routing protocols, you can filter the imported routes.

### Usage Scenario

After defining a filter (such as the IP prefix list, ACL, or route-policy) of a routing policy, you need to apply the filter to routing protocols.

You can apply routing policies when you want to import external routes:

- You can use the **import-route** command in the related protocol view, import the required external routes to the protocols, and apply a route-policy to filter imported routes.
- After external routes are imported, you can run the **filter-policy export** command to filter the imported external routes. Only the routes that meet the matching rules are advertised.

#### NOTE

- BGP has the powerful filtering function. For the configuration of BGP routing policies, refer to "BGP Configuration."
- For details of the **filter-policy** and **import-route** commands and their applications in RIP, OSPF, IS-IS, and BGP, refer to related configurations.

### Pre-configuration Tasks

Before applying filters to imported routes, complete the following tasks:

- [Configure an IP prefix list.](#)
- Configure an ACL.
- [Configure a route-policy.](#)

## Configuring RIP to Import External Routes

Routing Information Protocol (RIP) can import routes from other processes or other routing protocols to enrich the RIP routing table.

### Context

On a large-scale network, different routing protocols may be configured. In this situation, configure RIP to import routes from other processes or other routing protocols.

### Procedure

#### Step 1 Run **system-view**

The system view is displayed.

#### Step 2 Run **rip [ process-id ]**

A RIP process is created, and the RIP view is displayed.

#### Step 3 (Optional) Run **default-cost cost**

The default cost is set for imported routes.

If no cost is specified for imported external routes, the default cost 0 takes effect.

#### Step 4 Run **import-route**

External routes are imported.

- To import direct routes, static routes, IS-IS routes, OSPF routes, or routes of other RIP processes, run the **import-route { direct | static |{ isis | ospf | rip } [ process-id ] } [ cost cost | route-policy route-policy-name ] \* command.**
- To import IBGP routes, run the **import-route bgp permit-ibgp [ cost { cost | transparent } | route-policy route-policy-name ] \* \* command.**

#### NOTE

Importing routes from other protocols to a RIP process may lead to routing loops. Therefore, exercise caution when running the **import-route** command.

#### Step 5 Run **commit**

The configuration is committed.

----End

## Configuring OSPF to Import External Routes

Importing the routes discovered by other routing protocols can enrich OSPF routing information.

## Context

OSPF provides loop-free intra-area routes and inter-area routes; however, OSPF cannot prevent external routing loops. Therefore, you should exercise caution when configuring OSPF to import external routes.

Perform the following operations on the router that functions as the ASBR running OSPF:

## Procedure

- Configure OSPF to import the routes discovered by other protocols.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **ospf [ process-id ]**

The OSPF view is displayed.
  - c. Run **import-route { bgp [ permit-ibgp ] | direct | rip [ process-id-rip ] | static | isis [ process-id-isis ] | ospf [ process-id-ospf ] } [ cost cost | route-policy route-policy-name | tag tag | type type ]<sup>\*</sup>**

External routes are imported.

    - The **cost** *cost* parameter specifies the cost of a route.
    - The **type** *type* parameter specifies the type of a route. It can be 1 or 2.
    - The **tag** *tag* parameter specifies the tag in the external LSA.
    - The **route-policy** *route-policy-name* parameter specifies the name of a route-policy.
  - d. Run **commit**

The configuration is committed.
- Set parameters for OSPF to import routes.
  - a. Run **system-view**

The system view is displayed.
  - b. Run **ospf [ process-id ]**

The OSPF view is displayed.
  - c. Run **default { cost { cost | inherit-metric } | tag tag | type type }<sup>\*</sup>**

The default parameter (cost, tag, and type) values are configured for imported routes

    - The **cost** *cost* parameter specifies the default cost of the external routes imported by OSPF.
    - The **inherit-metric** parameter retains the original costs of the imported routes. If the cost is not specified, the default cost set through the **default** command is used as the cost of the imported routes.

When OSPF imports external routes, you can set default values for some additional parameters, such as the route cost, tag, and type. The route tag is used to identify the protocol-related information. For example, it can be used to differentiate AS numbers when OSPF imports BGP routes.

#### NOTE

You can run one of the following commands to set the cost of the imported routes. The following commands are listed in descending order of priority:

- Run the **apply cost** command in a route-policy to set the cost of the imported routes.
- Run the **import-route** command for OSPF to set the cost of the imported routes.
- Run the **default** command to set the default cost of the imported routes.

d. Run **commit**

The configuration is committed.

----End

## Configuring IS-IS to Import External Routes

By configuring IS-IS to import routes, you can enable IS-IS to learn routing information of other protocols or other IS-IS processes.

## Context

IS-IS regards the routes discovered by other routing protocols or other IS-IS processes as external routes. You can specify default costs for imported routes.

## Procedure

**Step 1** Run **system-view**

The system view is displayed.

**Step 2** Run **isis [ process-id ]**

The IS-IS view is displayed.

**Step 3** Configure IS-IS to import external routes.

Before setting costs for the imported routes, run the **import-route { direct | static | { ospf | rip | isis } [ process-id ] | bgp [ permit-ibgp ] } [ cost-type { external | internal } | cost cost | tag tag | route-policy route-policy | [ level-1 | level-2 | level-1-2 ] ] \* command** to import external routes.

Before retaining the original costs of the imported routes, run the **import-route { { ospf | rip | isis } [ process-id ] | bgp [ permit-ibgp ] | direct } inherit-cost [ { level-1 | level-2 | level-1-2 } | tag tag | route-policy route-policy ] \* command** to import external routes.

 NOTE

**permit-ibgp** takes effect only in the public network instance.

When the cost type of IS-IS is narrow, the parameters **cost-type { external | internal }** will determine the cost of imported routes.

- If you configure **external**, the cost of imported routes equals the original route cost plus 64.
- If you configure **internal**, the cost of imported routes is the same as the original route cost.

To better manage and maintain IS-IS networks, configure **route-policy route-policy** to import only required routes.

If you do not specify **level-1**, **level-2**, or **level-1-2** in the command, routes are imported to the Level-2 routing table by default.

**Step 4** Run **commit**

The configuration is committed.

----End

## Verifying the Configuration

After applying filters to the imported routes, check information about the routing table of each protocol.

## Prerequisites

Filters have been applied to the imported routes.

## Procedure

- Run the **display rip process-id route** command to check information about the RIP routing table.
- Run the **display ospf [ process-id ] routing** command to check information about the OSPF routing table.
- Run the **display isis [ process-id ] route** command to check information about the IS-IS routing table.
- Run the **display ip routing-table** command to check information about the IP routing table.

Run the **display ip routing-table** command on the local router, and you can view that the routes matching the filtering rules of the neighbor have been filtered out or the **apply** action has been performed.

----End

### 1.1.11.2.14 Configuration Examples for Routing Policies

This section provides some routing policy configuration examples.

## Example for Filtering the Routes to Be Received and Advertised

Filters can be applied to the routes to be received and advertised based on networking requirements.

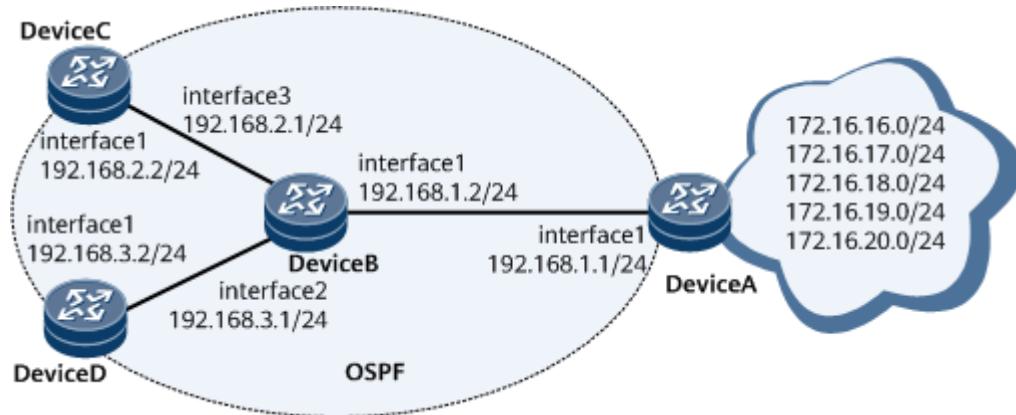
## Networking Requirements

On the OSPF network shown in **Figure 1-469**, DeviceA receives routes from the Internet and provides some of the Internet routes for DeviceB. It is required that DeviceA provide only routes 172.16.17.0/24, 172.16.18.0/24, and 172.16.19.0/24 to DeviceB, DeviceC accept only 172.16.18.0/24, and that DeviceD accept all the routes provided by DeviceB.

**Figure 1-469** Network diagram of filtering routes to be received and advertised

### NOTE

Interfaces 1 through 3 in this example represent GE1/0/0, GE2/0/0, and GE3/0/0, respectively.



## Precautions

During the configuration, pay attention to the following points:

- When configuring an IP prefix list, you need to specify the IP prefix range as required.
- The name of the IP prefix list to be referenced is case-sensitive.

## Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on DeviceA, DeviceB, DeviceC, and DeviceD.
2. Configure static routes on DeviceA and import them to the OSPF routing table.
3. Configure an export policy on DeviceA and check the filtering result on DeviceB.
4. Configure an import policy on DeviceC and check the filtering result on DeviceC.

## Data Preparation

To complete the configuration, you need the following data:

- Five static routes imported by DeviceA

- OSPF backbone area (area 0) where DeviceA, DeviceB, DeviceC, and DeviceD reside
- Name of the IP prefix list, which is used to filter routes

## Procedure

**Step 1** Assign an IP address to each interface.

**Step 2** Configure OSPF.

# Configure DeviceA.

```
[~DeviceA] ospf
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0.0] quit
[~DeviceA-ospf-1] quit
```

# Configure DeviceB.

```
[~DeviceB] ospf
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.2.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 192.168.3.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] commit
[~DeviceB-ospf-1-area-0.0.0.0] quit
```

# Configure DeviceC.

```
[~DeviceC] ospf
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 192.168.2.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
[~DeviceC-ospf-1] quit
```

# Configure DeviceD.

```
[~DeviceD] ospf
[*DeviceD-ospf-1] area 0
[*DeviceD-ospf-1-area-0.0.0.0] network 192.168.3.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0.0] commit
[~DeviceD-ospf-1-area-0.0.0.0] quit
```

**Step 3** Configure five static routes on DeviceA and import them to the OSPF routing table.

```
[~DeviceA] ip route-static 172.16.16.0 24 NULL0
[*DeviceA] ip route-static 172.16.17.0 24 NULL0
[*DeviceA] ip route-static 172.16.18.0 24 NULL0
[*DeviceA] ip route-static 172.16.19.0 24 NULL0
[*DeviceA] ip route-static 172.16.20.0 24 NULL0
[*DeviceA] ospf
[*DeviceA-ospf-1] import-route static
[*DeviceA-ospf-1] commit
[~DeviceA-ospf-1] quit
```

# Check the IP routing table on DeviceB. The command output shows the five static routes imported by OSPF.

```
[~DeviceB] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```

Routing Table : Public
Destinations : 22 Routes : 22
```

| Destination/Mask      | Proto        | Pre        | Cost     | Flags    | NextHop            | Interface                   |
|-----------------------|--------------|------------|----------|----------|--------------------|-----------------------------|
| 1.1.1.1/32            | Direct       | 0          | 0        | D        | 127.0.0.1          | LoopBack1                   |
| 127.0.0.0/8           | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| 127.0.0.1/32          | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| 127.255.255.255/32    | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| 255.255.255.255/32    | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| <b>172.16.16.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| <b>172.16.17.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| <b>172.16.18.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| <b>172.16.19.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| <b>172.16.20.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| 192.168.1.0/24        | Direct       | 0          | 0        | D        | 192.168.1.2        | GigabitEthernet1/0/0        |
| 192.168.1.1/32        | Direct       | 0          | 0        | D        | 192.168.1.1        | GigabitEthernet1/0/0        |
| 192.168.1.2/32        | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet1/0/0        |
| 192.168.1.255/32      | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet1/0/0        |
| 192.168.2.0/24        | Direct       | 0          | 0        | D        | 192.168.2.1        | GigabitEthernet3/0/0        |
| 192.168.2.1/32        | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet3/0/0        |
| 192.168.2.2/32        | Direct       | 0          | 0        | D        | 192.168.2.2        | GigabitEthernet3/0/0        |
| 192.168.2.255/32      | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet3/0/0        |
| 192.168.3.0/24        | Direct       | 0          | 0        | D        | 192.168.3.1        | GigabitEthernet2/0/0        |
| 192.168.3.1/32        | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet2/0/0        |
| 192.168.3.2/32        | Direct       | 0          | 0        | D        | 192.168.3.2        | GigabitEthernet2/0/0        |
| 192.168.3.255/32      | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet2/0/0        |

#### Step 4 Configure an export policy to filter routes to be advertised.

# Configure an IP prefix list named **a2b** on DeviceA.

```
[~DeviceA] ip ip-prefix a2b index 10 permit 172.16.17.0 24
[*DeviceA] ip ip-prefix a2b index 20 permit 172.16.18.0 24
[*DeviceA] ip ip-prefix a2b index 30 permit 172.16.19.0 24
[*DeviceA] commit
```

# Configure an export policy that is based on the IP prefix list **a2b** on DeviceA.

```
[~DeviceA] ospf
[*DeviceA-ospf-1] filter-policy ip-prefix a2b export static
[*DeviceA-ospf-1] commit
[~DeviceA-ospf-1] quit
```

# Check the IP routing table on DeviceB. The command output shows that DeviceB accepted only the three routes defined in IP prefix list **a2b**.

```
[~DeviceB] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

| Destination/Mask      | Proto        | Pre        | Cost     | Flags    | NextHop            | Interface                   |
|-----------------------|--------------|------------|----------|----------|--------------------|-----------------------------|
| 1.1.1.1/32            | Direct       | 0          | 0        | D        | 127.0.0.1          | LoopBack1                   |
| 127.0.0.0/8           | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| 127.0.0.1/32          | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| 127.255.255.255/32    | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| 255.255.255.255/32    | Direct       | 0          | 0        | D        | 127.0.0.1          | InLoopBack0                 |
| <b>172.16.17.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| <b>172.16.18.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| <b>172.16.19.0/24</b> | <b>O_ASE</b> | <b>150</b> | <b>1</b> | <b>D</b> | <b>192.168.1.1</b> | <b>GigabitEthernet1/0/0</b> |
| 192.168.1.0/24        | Direct       | 0          | 0        | D        | 192.168.1.2        | GigabitEthernet1/0/0        |
| 192.168.1.1/32        | Direct       | 0          | 0        | D        | 192.168.1.1        | GigabitEthernet1/0/0        |
| 192.168.1.2/32        | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet1/0/0        |
| 192.168.1.255/32      | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet1/0/0        |
| 192.168.2.0/24        | Direct       | 0          | 0        | D        | 192.168.2.1        | GigabitEthernet3/0/0        |
| 192.168.2.1/32        | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet3/0/0        |
| 192.168.2.2/32        | Direct       | 0          | 0        | D        | 192.168.2.2        | GigabitEthernet3/0/0        |
| 192.168.2.255/32      | Direct       | 0          | 0        | D        | 127.0.0.1          | GigabitEthernet3/0/0        |
| 192.168.3.0/24        | Direct       | 0          | 0        | D        | 192.168.3.1        | GigabitEthernet2/0/0        |

```
192.168.3.1/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
192.168.3.2/32 Direct 0 0 D 192.168.3.2 GigabitEthernet2/0/0
192.168.3.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet2/0/0
```

### Step 5 Configure an import policy.

```
Configure an IP prefix list named in on DeviceC.
```

```
[~DeviceC] ip ip-prefix in index 10 permit 172.16.18.0 24
[*DeviceC] commit
```

```
Configure an import policy that is based on the IP prefix list in on DeviceC.
```

```
[~DeviceC] ospf
[*DeviceC-ospf-1] filter-policy ip-prefix in import
[*DeviceC-ospf-1] commit
```

```
Check the IP routing table on DeviceC. The command output shows that
DeviceC accepted only one route (the one matching IP prefix list in).
```

```
[~DeviceC] display ip routing-table
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
```

```
Routing Table : Public
Destinations : 12 Routes : 12
Destination/Mask Proto Pre Cost Flags NextHop Interface
1.1.1.1/32 O_ASE 10 1 D 192.168.2.1 GigabitEthernet1/0/0
127.0.0.0/8 Direct 0 0 D 127.0.0.1 InLoopBack0
127.0.0.1/32 Direct 0 0 D 127.0.0.1 InLoopBack0
127.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
255.255.255.255/32 Direct 0 0 D 127.0.0.1 InLoopBack0
172.16.18.0/24 O_ASE 150 1 D 192.168.2.1 GigabitEthernet1/0/0
192.168.1.0/24 O_ASE 10 2 D 192.168.2.1 GigabitEthernet1/0/0
192.168.2.0/24 Direct 0 0 D 192.168.2.2 GigabitEthernet1/0/0
192.168.2.1/32 Direct 0 0 D 192.168.2.1 GigabitEthernet1/0/0
192.168.2.2/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
192.168.2.255/32 Direct 0 0 D 127.0.0.1 GigabitEthernet1/0/0
192.168.3.0/24 O_ASE 10 2 D 192.168.2.1 GigabitEthernet1/0/0
```

----End

## Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.1 255.255.255.0
#
ospf 1
filter-policy ip-prefix a2b export static
import-route static
area 0.0.0.0
network 192.168.1.0 0.0.0.255
#
ip ip-prefix a2b index 10 permit 172.16.17.0 24
ip ip-prefix a2b index 20 permit 172.16.18.0 24
ip ip-prefix a2b index 30 permit 172.16.19.0 24
#
ip route-static 172.16.16.0 255.255.255.0 NULL0
ip route-static 172.16.17.0 255.255.255.0 NULL0
ip route-static 172.16.18.0 255.255.255.0 NULL0
ip route-static 172.16.19.0 255.255.255.0 NULL0
ip route-static 172.16.20.0 255.255.255.0 NULL0
```

- ```
#  
return
```
- DeviceB configuration file

```
#  
sysname DeviceB  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.1.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
undo shutdown  
ip address 192.168.3.1 255.255.255.0  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 192.168.2.1 255.255.255.0  
#  
ospf 1  
area 0.0.0.  
network 192.168.1.0 0.0.0.255  
network 192.168.2.0 0.0.0.255  
network 192.168.3.0 0.0.0.255  
#  
return
```
 - DeviceC configuration file

```
#  
sysname DeviceC  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.2.2 255.255.255.0  
#  
ospf 1  
filter-policy ip-prefix in import  
area 0.0.0.  
network 192.168.2.0 0.0.0.255  
#  
ip ip-prefix in index 10 permit 172.16.18.0 24  
#  
return
```
 - DeviceD configuration file

```
#  
sysname DeviceD  
#  
interface GigabitEthernet1/0/0  
undo shutdown  
ip address 192.168.3.2 255.255.255.0  
#  
ospf 1  
area 0.0.0.  
network 192.168.3.0 0.0.0.255  
#  
return
```

Example for Applying a Route-Policy to Route Import

By applying route-policies, you can control the import of routes and set attributes for imported routes.

Networking Requirements

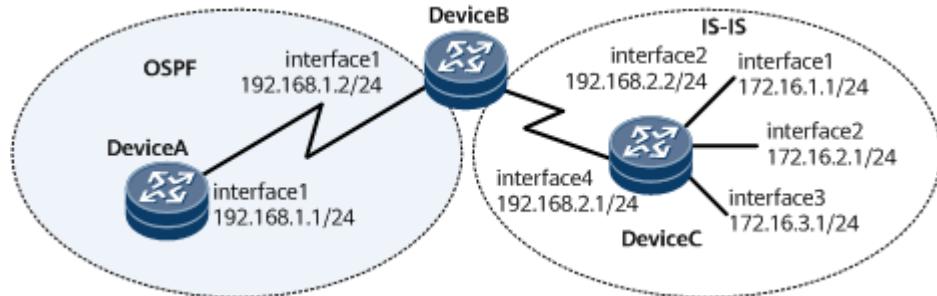
On the network shown in [Figure 1-470](#), DeviceB exchanges routing information with DeviceA through OSPF and with DeviceC through IS-IS.

It is required that IS-IS routes be imported into OSPF on DeviceB and that a route-policy be used to set route attributes. Specifically, the cost of the route 172.16.1.0/24 is set to 100, and the tag of the route 172.16.2.0/24 is set to 20.

Figure 1-470 Applying a route-policy to route import

 **NOTE**

In this example, interface1, interface2, interface3, and interface4 represent GE1/0/0, GE2/0/0, GE3/0/0, and GE1/0/1, respectively.



Precautions

During the configuration, pay attention to the following points:

- When configuring an IP prefix list, specify a proper IP prefix range according to actual requirements.
- The name of a route-policy is case sensitive.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic IS-IS functions on DeviceB and DeviceC.
2. Configure OSPF on DeviceA and DeviceB, and import IS-IS routes into OSPF.
3. Configure a route-policy on DeviceB, apply the route-policy to route import from IS-IS to OSPF, and check routing information.

Data Preparation

To complete the configuration, you need the following data:

- Area IDs, IS-IS levels, and system IDs of DeviceB and DeviceC
- Area IDs (0, indicating the OSPF backbone area) of DeviceA and DeviceB
- ACL number, name of the IP prefix list, cost of the route 172.16.1.0/24, and tag of the route 172.16.2.0/24

Procedure

Step 1 Configure IP addresses for interfaces. For detailed configurations, see Configuration Files.

Step 2 Configure IS-IS.

Configure DeviceC.

```
[~DeviceC] isis
[*DeviceC-isis-1] is-level level-2
[*DeviceC-isis-1] network-entity 10.0000.0000.0001.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 1/0/1
[*DeviceC-GigabitEthernet1/0/1] isis enable
[*DeviceC-GigabitEthernet1/0/1] quit
[*DeviceC] interface GigabitEthernet 1/0/0
[*DeviceC-GigabitEthernet1/0/0] isis enable
[*DeviceC-GigabitEthernet1/0/0] quit
[*DeviceC] interface GigabitEthernet 2/0/0
[*DeviceC-GigabitEthernet2/0/0] isis enable
[*DeviceC-GigabitEthernet2/0/0] quit
[*DeviceC] interface GigabitEthernet 3/0/0
[*DeviceC-GigabitEthernet3/0/0] isis enable
[*DeviceC-GigabitEthernet3/0/0] commit
[~DeviceC-GigabitEthernet3/0/0] quit
```

Configure DeviceB.

```
[~DeviceB] isis
[*DeviceB-isis-1] is-level level-2
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 2/0/0
[*DeviceB-GigabitEthernet2/0/0] isis enable
[*DeviceB-GigabitEthernet2/0/0] commit
[~DeviceB-GigabitEthernet2/0/0] quit
```

Step 3 Configure OSPF and import routes.

Enable OSPF on DeviceA.

```
[~DeviceA] ospf
[*DeviceA-ospf-1] area 0
[*DeviceA-ospf-1-area-0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0] quit
[~DeviceA-ospf-1] quit
```

Enable OSPF and import IS-IS routes on DeviceB.

```
[~DeviceB] ospf
[*DeviceB-ospf-1] area 0
[*DeviceB-ospf-1-area-0.0.0] network 192.168.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0] quit
[*DeviceB-ospf-1] import-route isis 1
[*DeviceB-ospf-1] commit
[~DeviceB-ospf-1] quit
```

Check the OSPF routing table on DeviceA. The command output shows the imported routes.

```
[~DeviceA] display ospf routing
    OSPF Process 1 with Router ID 192.168.1.1
        Routing Tables
        Routing for Network
        Destination Cost Type NextHop AdvRouter Area
        192.168.1.0/24 1 Stub 192.168.1.1 192.168.1.1 0.0.0.0
        Routing for ASEs
        Destination Cost Type Tag NextHop AdvRouter
        172.16.1.0/24 1 Type2 1 192.168.1.2 192.168.1.2
        172.16.2.0/24 1 Type2 1 192.168.1.2 192.168.1.2
        172.16.3.0/24 1 Type2 1 192.168.1.2 192.168.1.2
        192.168.2.0/24 1 Type2 1 192.168.1.2 192.168.1.2
        Routing for NSSAs
        Destination Cost Type Tag NextHop AdvRouter
        Total Nets: 5
        Intra Area: 1 Inter Area: 0 ASE: 4 NSSA: 0
```

Step 4 Configure filters.

Configure an ACL numbered **2002** to permit the route 172.16.2.0/24.

```
[~DeviceB] acl number 2002
[*DeviceB-acl4-basic-2002] rule permit source 172.16.2.0 0.0.0.255
[*DeviceB-acl4-basic-2002] commit
[~DeviceB-acl4-basic-2002] quit
```

Configure an IP prefix list named **prefix-a** to permit the route 172.16.1.0/24.

```
[~DeviceB] ip ip-prefix prefix-a index 10 permit 172.16.1.0 24
[*DeviceB] commit
```

Step 5 Configure a route-policy.

```
[~DeviceB] route-policy isis2ospf permit node 10
[*DeviceB-route-policy] if-match ip-prefix prefix-a
[*DeviceB-route-policy] apply cost 100
[*DeviceB-route-policy] quit
[*DeviceB] route-policy isis2ospf permit node 20
[*DeviceB-route-policy] if-match acl 2002
[*DeviceB-route-policy] apply tag 20
[*DeviceB-route-policy] quit
[*DeviceB] route-policy isis2ospf permit node 30
[*DeviceB] commit
[~DeviceB-route-policy] quit
```

Step 6 Apply the route-policy to route import.

Apply the route-policy to route import on DeviceB.

```
[~DeviceB] ospf
[*DeviceB-ospf-1] import-route isis 1 route-policy isis2ospf
[*DeviceB-ospf-1] commit
[~DeviceB-ospf-1] quit
```

Check the OSPF routing table on DeviceA. The command output shows that the cost of the route 172.16.1.0/24 is 100, the tag of the route 172.16.2.0/24 is 20, and that attributes of other routes remain unchanged.

```
[~DeviceA] display ospf routing
      OSPF Process 1 with Router ID 192.168.1.1
      Routing Tables
      Routing for Network
      Destination   Cost   Type     NextHop     AdvRouter   Area
      192.168.1.0/24    1  Stub      192.168.1.1  192.168.1.1  0.0.0.0
      Routing for ASEs
      Destination   Cost   Type     Tag       NextHop     AdvRouter
      172.16.1.0/24   100   Type2    1        192.168.1.2  192.168.1.2
      172.16.2.0/24    1   Type2    20      192.168.1.2  192.168.1.2
      172.16.3.0/24    1   Type2    1        192.168.1.2  192.168.1.2
      192.168.2.0/24   1   Type2    1        192.168.1.2  192.168.1.2
      Routing for NSSAs
      Destination   Cost   Type     Tag       NextHop     AdvRouter
      Total Nets: 5
      Intra Area: 1 Inter Area: 0 ASE: 4 NSSA: 0
```

----End

Configuration Files

- DeviceA configuration file

```
# 
sysname DeviceA
#
interface GigabitEthernet1/0/0
undo shutdown
```

```
ip address 192.168.1.1 255.255.255.0
#
ospf 1
area 0.0.0
network 192.168.1.0 0.0.0.255
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
acl number 2002
rule 5 permit source 172.16.2.0 0.0.0.255
#
isis 1
is-level level-2
network-entity 10.0000.0000.0002.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 192.168.2.2 255.255.255.0
isis enable 1
#
ospf 1
import-route isis 1 route-policy isis2ospf
area 0.0.0
network 192.168.1.0 0.0.0.255
#
route-policy isis2ospf permit node 10
if-match ip-prefix prefix-a
apply cost 100
#
route-policy isis2ospf permit node 20
if-match acl 2002
apply tag 20
#
route-policy isis2ospf permit node 30
#
ip ip-prefix prefix-a index 10 permit 172.16.1.0 24
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
isis 1
is-level level-2
network-entity 10.0000.0000.0001.00
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 172.16.1.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet2/0/0
undo shutdown
ip address 172.16.2.1 255.255.255.0
isis enable 1
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 172.16.3.1 255.255.255.0
isis enable 1
#
```

```
interface GigabitEthernet1/0/1
undo shutdown
ip address 192.168.2.1 255.255.255.0
isis enable 1
#
return
```

Example for Applying Route-Policies

By configuring route-policies, you can flexibly control the traffic on a complex network.

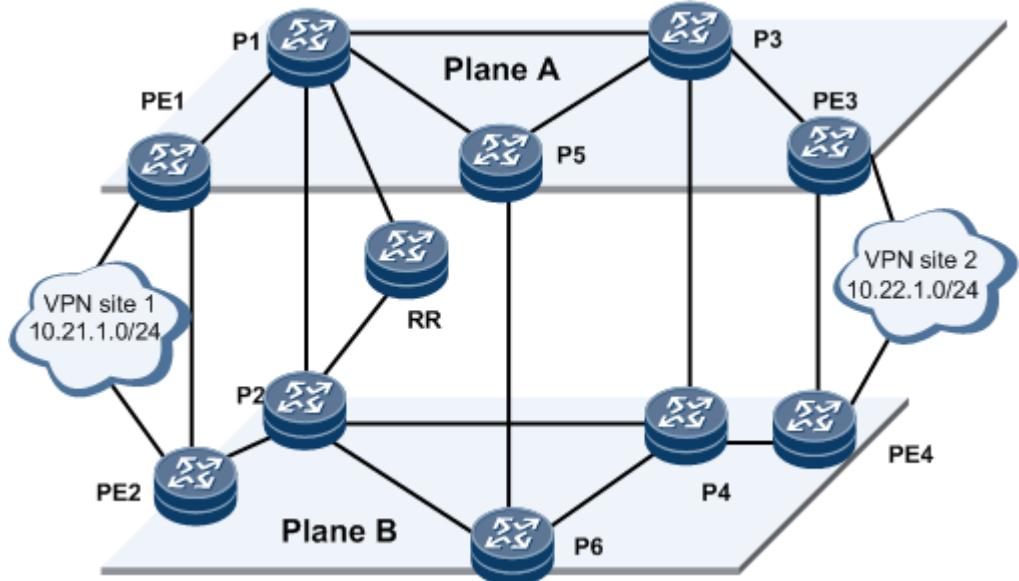
Networking Requirements

Figure 1-471 shows a simplified diagram of the MPLS network that carries multiple types of L3VPN services, such as multimedia, signaling, and accounting. In **Figure 1-471**, two sites, each of which has two PEs accessing the core layer, are used as an example. The core layer is divided into two planes, each of which has three fully meshed P nodes. Nodes in different planes are connected to provide backup paths. MP-BGP is used to advertise inner tags and VPNv4 routes between the PEs. Each PE needs to establish the MP-IBGP peer relationship with a route reflector (RR).

NOTE

Figure 1-471 is a simplified networking diagram, in which there are two sites, one RR, and two planes with six P nodes. On the actual network, there are 14 sites with 28 PEs, and each plane has four P nodes and two RRs. Therefore, each RR needs to establish MP-IBGP connections with 28 PEs.

Figure 1-471 Applying route-policies



In **Figure 1-471**, each PE sends BGP Update messages to the RR, and the other PEs receive BGP Update messages from different planes. Therefore, route-policies need to be applied to ensure that a VPN traffic flow is transmitted through only one plane.

Precautions

When applying route-policies, note the following rules:

- Configure different route distinguishers (RDs) for the two PEs in the same site.
- Assign different community attributes for the routes advertised by PEs in different planes.
- Run the **undo policy vpn-target** command in the BGP-VPNv4 address family view to ensure that VPN-target-based filtering is not performed on VPNv4 routes.
- When applying a routing policy, ensure that the routing policy name is case sensitive.

Configuration Roadmap

The configuration roadmap is as follows:

- Configure different RDs for two PEs in the same site to ensure that each PE can receive two routes from different BGP next hops in the remote site. When two PEs in a site advertise the routes to the same destination, configuring different RDs for the two PEs can ensure that BGP peers consider the advertised routes as two different routes. This is because BGP-VPNv4 uses the VPNv4 addresses that consist of IPv4 addresses and RDs.
- Assign different community attributes to the routes advertised by the PE in plane A and the routes advertised by the PE in plane B.
- Set different local priorities for routes based on the community attributes of the routes. In this manner, the PE in plane A preferentially selects the routes advertised by the remote PE in plane A, and the PE in plane B preferentially selects the routes advertised by the remote PE in plane B.

Data Preparation

To complete the configuration, you need the following data.

Table 1-176 IP addresses of physical interfaces

Local Device	Local Interface and Its IP Address	Remote Interface and Its IP Address	Remote Device
P1	GE 1/0/0 10.1.1.1/30	GE 1/0/0 10.1.1.2/30	P3
P1	GE 2/0/0 10.1.2.1/30	GE 1/0/0 10.1.2.2/30	P5
P1	GE 3/0/0 10.1.3.1/30	GE 1/0/0 10.1.3.2/30	RR
P1	GE 1/0/1 10.1.4.1/30	GE 1/0/0 10.1.4.2/30	P2

Local Device	Local Interface and Its IP Address	Remote Interface and Its IP Address	Remote Device
P1	GE 1/0/2 10.1.5.1/30	GE 1/0/0 10.1.5.2/30	PE1
P2	GE 1/0/1 10.1.6.1/30	GE 1/0/0 10.1.6.2/30	P6
P2	GE 3/0/0 10.1.7.1/30	GE 1/0/0 10.1.7.2/30	P4
P2	GE 2/0/0 10.1.8.1/30	GE 2/0/0 10.1.8.2/30	RR
P2	GE 1/0/2 10.1.9.1/30	GE 1/0/0 10.1.9.2/30	PE2
P3	GE 2/0/0 10.1.10.1/30	GE 2/0/0 10.1.10.2/30	P5
P3	GE 3/0/0 10.1.11.1/30	GE 2/0/0 10.1.11.2/30	P4
P3	GE 1/0/1 10.1.12.1/30	GE 1/0/0 10.1.12.2/30	PE3
P4	GE 3/0/0 10.1.13.1/30	GE 3/0/0 10.1.13.2/30	P6
P4	GE 1/0/1 10.1.14.1/30	GE 1/0/0 10.1.14.2/30	PE4
P5	GE 3/0/0 10.1.15.1/30	GE 2/0/0 10.1.15.2/30	P6
PE1	GE 2/0/0 10.1.16.1/30	GE 2/0/0 10.1.16.2/30	PE2
PE3	GE 2/0/0 10.1.17.1/30	GE 2/0/0 10.1.17.2/30	PE4

Table 1-177 IP addresses of loopback interfaces

Local Device	IP Address of Local Loopback 0 Interface
P1	10.1.1.9/32
P2	10.2.2.9/32

Local Device	IP Address of Local Loopback 0 Interface
P3	10.3.3.9/32
P4	10.4.4.9/32
P5	10.5.5.9/32
P6	10.6.6.9/32
PE1	10.7.7.9/32
PE2	10.8.8.9/32
PE3	10.9.9.9/32
PE4	10.10.10.9/32
RR	10.11.11.9/32

Table 1-178 BGP parameter values

BGP Parameter	Value
AS number	65000
Router ID	Same as the address of Loopback 0 interface
BGP community attribute	Plane A: 65000:100 Plane B: 65000:200
BGP local priority	Plane A: The local priority of community attribute 65000:100 is set to 200. Plane B: The local priority of community attribute 65000:200 is set to 200.
Routing policy name	Policy to filter imported routes: local_pre Policy to filter routes to be advertised: comm
Community filter name	1
BGP peer group name	Client

Procedure

Step 1 Configure names for devices and IP addresses for interfaces.

For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure an IGP.

In this example, IS-IS is used. For detailed configurations, see the configuration files in this example.

After the configuration, run the **display ip routing-table** command. You can view that PEs, P nodes and PEs, and P nodes have learned the addresses of Loopback 0 interfaces from each other.

Step 3 Establish MP-IBGP connections between PEs and RRs.

Use the configuration of PE1 as an example. Configurations of other PEs are similar to that of PE1. For configuration details, see [Configuration Files](#) in this section.

```
[*PE1] bgp 65000
[~PE1-bgp] peer 10.11.11.9 as-number 65000
[*PE1-bgp] peer 10.11.11.9 connect-interface LoopBack0
[*PE1-bgp] ipv4-family unicast
[*PE1-bgp-af-ipv4] undo peer 10.11.11.9 enable
[*PE1-bgp] ipv4-family vpngv4
[*PE1-bgp-af-vpngv4] peer 10.11.11.9 enable
[*PE1-bgp-af-vpngv4] commit
```

Configure the RR.

```
[~RR] bgp 65000
[*RR-bgp] group client internal
[*RR-bgp] peer client connect-interface LoopBack0
[*RR-bgp] ipv4-family unicast
[*RR-bgp-af-ipv4] undo peer client enable
[*RR-bgp-af-ipv4] quit
[*RR-bgp] ipv4-family vpngv4
[*RR-bgp-af-vpngv4] undo policy vpn-target
[*RR-bgp-af-vpngv4] peer client enable
[*RR-bgp-af-vpngv4] peer 10.7.7.9 group client
[*RR-bgp-af-vpngv4] peer 10.8.8.9 group client
[*RR-bgp-af-vpngv4] peer 10.9.9.9 group client
[*RR-bgp-af-vpngv4] peer 10.10.10.9 group client
[*RR-bgp-af-vpngv4] peer client reflect-client
[*RR-bgp-af-vpngv4] commit
[~RR-bgp-af-vpngv4] quit
```

NOTE

The **undo policy vpn-target** command needs to be run in the BGP-VPNv4 address family view of the RR to ensure that VPN-target-based filtering is not performed on VPNv4 routes. By default, an RR performs VPN-target-based filtering on the received VPNv4 routes. The routes that match the filtering rules are added to the VPN routing table, and other routes are discarded. In this example, no VPN instances are configured on the RR. In this case, if VPN-target-based filtering is enabled, all the received VPNv4 routes will be discarded.

After the configuration, run the **display bgp vpngv4 all peer** command on the RR. The following command output shows that the RR has established MP-IBGP connections with all PEs.

```
<RR> display bgp vpngv4 all peer
BGP local router ID : 10.11.11.9
Local AS number : 65000
Total number of peers : 4          Peers in established state : 4
Peer      V  AS  MsgRcvd  MsgSent  OutQ  Up/Down    State     PrefRcv
10.7.7.9  4  65000  79    82      0    00:01:31  Established  0
10.8.8.9  4  65000  42    66      0    00:01:16  Established  0
10.9.9.9  4  65000  21    34      0    00:00:50  Established  0
10.10.10.9 4  65000  2     4      0    00:00:21  Established  0
```

Step 4 Configure route-policies.

NOTE

Use the configurations of PE1, PE2, and the RR as an example. The configurations of PE3 and PE4 are similar to the configurations of PE1 and PE2 respectively. For configuration details, see [Configuration Files](#) in this section.

Configure a route-policy on PE1 so that the route advertised by the PE in plane A to the RR can carry community attribute 65000:100.

```
[~PE1] route-policy comm permit node 10
[*PE1-route-policy] apply community 65000:100
[*PE1-route-policy] commit
```

Configure a route-policy on PE2 so that the route advertised by the PE in plane B to the RR can carry community attribute 65000:200.

```
[~PE2] route-policy comm permit node 10
[*PE2-route-policy] apply community 65000:200
[*PE2-route-policy] commit
```

On PE1, apply the route-policy to the advertised BGP VPNv4 route so that the community attribute can be advertised to the RR.

```
[~PE1] bgp 65000
[*PE1-bgp] ipv4-family vpnv4
[*PE1-bgp-af-vpnv4] peer 10.11.11.9 route-policy comm export
[*PE1-bgp-af-vpnv4] peer 10.11.11.9 advertise-community
[*PE1-bgp-af-vpnv4] commit
```

On PE2, apply the route-policy to the advertised BGP VPNv4 route so that the community attribute can be advertised to the RR.

```
[~PE2] bgp 65000
[*PE2-bgp] ipv4-family vpnv4
[*PE2-bgp-af-vpnv4] peer 10.11.11.9 route-policy comm export
[*PE2-bgp-af-vpnv4] peer 10.11.11.9 advertise-community
[*PE2-bgp-af-vpnv4] commit
```

Configure the RR to advertise the community attributes to the PEs.

```
[~RR] bgp 65000
[*RR-bgp] ipv4-family vpnv4
[*RR-bgp-af-vpnv4] peer client advertise-community
[*RR-bgp-af-vpnv4] commit
```

Configure a community filter on PE1.

```
[~PE1] ip community-filter 1 permit 65000:100
[*PE1] commit
```

Configure a community filter on PE2.

```
[~PE2] ip community-filter 1 permit 65000:200
[*PE2] commit
```

On PE1, configure a route-policy and set the local preference of the route with community attribute 65000:100 to 200.

```
[~PE1] route-policy local_pre permit node 10
[*PE1-route-policy] if-match community-filter 1
[*PE1-route-policy] apply local-preference 200
[*PE1-route-policy] commit
[~PE1-route-policy] quit
```

On PE2, configure a route-policy and set the local preference of the route with community attribute 65000:200 to 200.

```
[~PE2] route-policy local_pre permit node 10
[*PE2-route-policy] if-match community-filter 1
[*PE2-route-policy] apply local-preference 200
[*PE2-route-policy] commit
[~PE2-route-policy] quit
```

On PE1, apply the route-policy to the imported BGP VPNv4 route so that the PE in plane A prefers the route advertised by the remote PE in plane A.

```
[~PE1] bgp 65000
[*PE1-bgp] ipv4-family vpnv4
[*PE1-bgp-af-vpnv4] peer 10.11.11.9 route-policy local_pre import
[*PE1-bgp-af-vpnv4] commit
```

On PE2, apply the route-policy to the imported BGP VPNv4 route so that the PE in plane B prefers the route advertised by the remote PE in plane B.

```
[~PE2] bgp 65000
[*PE2-bgp] ipv4-family vpnv4
[*PE2-bgp-af-vpnv4] peer 10.11.11.9 route-policy local_pre import
[*PE2-bgp-af-vpnv4] commit
```

NOTE

After this configuration, you also need to configure MPLS, establish tunnels, configure MPLS L3VPN functions, and connect the PEs to CEs. For configuration details, see [Configuration Files](#) in this section.

Step 5 Verify the configuration.

Run the **display bgp vpnv4 all routing-table community** command on a PE to view information about the VPNv4 routes with community attributes. Use the command output on PE1 and PE2 as an example.

```
[~PE1] display bgp vpnv4 all routing-table community
Total Number of Routes from all PE: 2
BGP Local router ID is 10.7.7.9
Status codes: * - valid, > - best, d - damped,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete

Route Distinguisher: 65000:10001012
      Network      NextHop      MED      LocPrf  PrefVal Community
*>  10.22.1.0/24  10.9.9.9      0      200    65000:100
*       10.10.10.9      0      100    65000:200
```

```
Total routes of vpn-instance NGN_Media: 2
      Network      NextHop      MED      LocPrf  PrefVal Community
*>i 10.22.1.0/24  10.9.9.9      0      200    0      65000:100
*       10.10.10.9      0      100    0      65000:200
```

```
[~PE2] display bgp vpnv4 all routing-table community
Total Number of Routes from all PE: 2
BGP Local router ID is 10.8.8.9
Status codes: * - valid, > - best, d - damped,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
```

```
Route Distinguisher: 65000:10001011
      Network      NextHop      MED      LocPrf  PrefVal Community
*>  10.22.1.0/24  10.10.10.9      0      200    65000:200
*       10.9.9.9      0      100    65000:100
```

```
Total routes of vpn-instance NGN_Media: 2
      Network      NextHop      MED      LocPrf  PrefVal Community
*>i 10.22.1.0/24  10.10.10.9      0      200    0      65000:200
*       10.9.9.9      0      100    0      65000:100
```

Run the **display ip routing-table vpn-instance NGN_Media 10.22.1.0 24** command on PE1. The command output shows that the next hop of the route to 10.22.1.0/24 is PE3, indicating that PE1 prefers the route advertised by PE3.

```
[~PE1] display ip routing-table vpn-instance NGN_Media 10.22.1.0 24
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
-----
Routing Table: NGN_Media
Destination/Mask Proto Pre Cost Flags NextHop Interface
10.22.1.0/24 BGP 255 0 RD 10.9.9.9 GigabitEthernet1/0/0
```

----End

Configuration Files

- P1 configuration file

```
#
sysname P1
#
mpls lsr-id 10.1.1.9
#
mpls
#
mpls ldp
#
isis 64
network-entity 49.0091.0100.0100.1009.00
#
interface GigabitEthernet1/0/0
description toP3GE1/0/0
undo shutdown
ip address 10.1.1.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet2/0/0
description toP5GE1/0/0
undo shutdown
ip address 10.1.2.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
description toRRGE1/0/0
undo shutdown
ip address 10.1.3.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet1/0/1
description toP2GE1/0/0
undo shutdown
ip address 10.1.4.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet1/0/2
description toP2GE1/0/0
undo shutdown
ip address 10.1.5.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
```

```
interface LoopBack0
ip address 10.1.1.9 255.255.255.255
isis enable 64
#
return
```

- P2 configuration file

```
# 
sysname P2
#
mpls lsr-id 10.2.2.9
#
mpls
#
mpls ldp
#
isis 64
network-entity 49.0091.0100.0200.2009.00
#
interface GigabitEthernet1/0/0
description toP1GE1/0/1
undo shutdown
ip address 10.1.4.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet2/0/0
description toRRGE2/0/0
undo shutdown
ip address 10.1.8.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
description toP4GE1/0/0
undo shutdown
ip address 10.1.7.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet1/0/1
description toP6GE1/0/0
undo shutdown
ip address 10.1.6.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet1/0/2
description toPE2GE1/0/0
undo shutdown
ip address 10.1.9.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface LoopBack0
ip address 10.2.2.9 255.255.255.255
isis enable 64
#
return
```

- P3 configuration file

```
# 
sysname P3
#
mpls lsr-id 10.3.3.9
```

```
#  
mpls  
#  
mpls ldp  
#  
isis 64  
network-entity 49.0091.0100.0300.3009.00  
#  
interface GigabitEthernet1/0/0  
description toP1GE1/0/0  
undo shutdown  
ip address 10.1.1.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet2/0/0  
description toP5GE2/0/0  
undo shutdown  
ip address 10.1.10.1 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet3/0/0  
description toP4GE2/0/0  
undo shutdown  
ip address 10.1.11.1 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet1/0/1  
description toPE3GE1/0/0  
undo shutdown  
ip address 10.1.12.1 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface LoopBack0  
ip address 10.3.3.9 255.255.255.255  
isis enable 64  
#  
return
```

- P4 configuration file

```
#  
sysname P4  
#  
mpls lsr-id 10.4.4.9  
#  
mpls  
#  
mpls ldp  
#  
isis 64  
network-entity 49.0091.0100.0400.4009.00  
#  
interface GigabitEthernet1/0/0  
description toP2GE3/0/0  
undo shutdown  
ip address 10.1.7.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet2/0/0  
description toP3GE3/0/0  
undo shutdown
```

```
ip address 10.1.11.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
description toP6GE3/0/0
undo shutdown
ip address 10.1.13.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet1/0/1
description toPE4GE1/0/0
undo shutdown
ip address 10.1.14.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface LoopBack0
ip address 10.4.4.9 255.255.255.255
isis enable 64
#
return
```

- P5 configuration file

```
#  
sysname P5  
#  
mpls lsr-id 10.5.5.9  
#  
mpls  
#  
mpls ldp  
#  
isis 64  
network-entity 49.0091.0100.0500.5009.00  
#  
interface GigabitEthernet1/0/0
description toP1GE2/0/0
undo shutdown
ip address 10.1.2.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet2/0/0
description toP3GE2/0/0
undo shutdown
ip address 10.1.10.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
description toP6GE2/0/0
undo shutdown
ip address 10.1.15.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface LoopBack0
ip address 10.5.5.9 255.255.255.255
isis enable 64
#
return
```

- P6 configuration file

```
#  
sysname P6  
#  
mpls lsr-id 10.6.6.9  
#  
mpls  
#  
mpls ldp  
#  
isis 64  
network-entity 49.0091.0100.0600.6009.00  
#  
interface GigabitEthernet1/0/0  
description toP2GE1/0/1  
undo shutdown  
ip address 10.1.6.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet2/0/0  
description toP5GE3/0/0  
undo shutdown  
ip address 10.1.15.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet3/0/0  
description toP4GE3/0/0  
undo shutdown  
ip address 10.1.13.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface LoopBack0  
ip address 10.6.6.9 255.255.255.255  
isis enable 64  
#  
return
```

- PE1 configuration file

```
#  
sysname PE1  
#  
ip vpn-instance NGN_Media  
route-distinguisher 65000:10001012  
apply-label per-instance  
vpn-target 65000:100 export-extcommunity  
vpn-target 65000:100 import-extcommunity  
vpn-target 65000:200 import-extcommunity  
vpn-target 65000:300 import-extcommunity  
ip vpn-instance NGN_Other  
route-distinguisher 65000:30001012  
apply-label per-instance  
vpn-target 65000:300 export-extcommunity  
vpn-target 65000:100 import-extcommunity  
vpn-target 65000:200 import-extcommunity  
vpn-target 65000:300 import-extcommunity  
ip vpn-instance NGN_Signaling  
route-distinguisher 65000:20001012  
apply-label per-instance  
vpn-target 65000:200 export-extcommunity  
vpn-target 65000:100 import-extcommunity  
vpn-target 65000:200 import-extcommunity  
vpn-target 65000:300 import-extcommunity  
#
```

```
mpls lsr-id 10.7.7.9
#
mpls
#
mpls ldp
#
isis 64
network-entity 49.0091.0100.0700.7009.00
#
interface GigabitEthernet1/0/0
description toP1GE1/0/2
undo shutdown
ip address 10.1.5.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet2/0/0
description toPE2GE2/0/0
undo shutdown
ip address 10.1.16.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
undo shutdown
#
interface GigabitEthernet3/0/0.10
ip binding vpn-instance NGN_Media
vlan-type dot1q 10
ip address 10.21.1.73 255.255.255.252
#
interface GigabitEthernet3/0/0.11
ip binding vpn-instance NGN_Signaling
vlan-type dot1q 11
ip address 10.21.1.77 255.255.255.252
#
interface GigabitEthernet3/0/0.12
ip binding vpn-instance NGN_Other
vlan-type dot1q 12
ip address 10.21.1.81 255.255.255.252
#
interface LoopBack0
ip address 10.7.7.9 255.255.255.255
isis enable 64
#
bgp 65000
peer 10.11.11.9 as-number 65000
peer 10.11.11.9 connect-interface LoopBack0
#
ipv4-family unicast
undo synchronization
undo peer 10.11.11.9 enable
#
ipv4-family vpng4
policy vpn-target
peer 10.11.11.9 enable
peer 10.11.11.9 route-policy local_pre import
peer 10.11.11.9 route-policy comm export
peer 10.11.11.9 advertise-community
#
ipv4-family vpn-instance NGN_Media
aggregate 10.21.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Other
aggregate 10.21.1.0 255.255.255.0 detail-suppressed
import-route direct
```

```
#  
ipv4-family vpn-instance NGN_Signaling  
aggregate 10.21.1.0 255.255.255.0 detail-suppressed  
import-route direct  
#  
route-policy comm permit node 10  
apply community 65000:100  
#  
ip community-filter 1 permit 65000:100  
#  
route-policy local_pre permit node 10  
if-match community-filter 1  
apply local-preference 200  
#  
return
```

- PE2 configuration file

```
#  
sysname PE2  
#  
ip vpn-instance NGN_Media  
route-distinguisher 65000:10001011  
apply-label per-instance  
vpn-target 65000:100 export-extcommunity  
vpn-target 65000:100 import-extcommunity  
vpn-target 65000:200 import-extcommunity  
vpn-target 65000:300 import-extcommunity  
ip vpn-instance NGN_Other  
route-distinguisher 65000:30001011  
apply-label per-instance  
vpn-target 65000:300 export-extcommunity  
vpn-target 65000:100 import-extcommunity  
vpn-target 65000:200 import-extcommunity  
vpn-target 65000:300 import-extcommunity  
ip vpn-instance NGN_Signaling  
route-distinguisher 65000:20001011  
apply-label per-instance  
vpn-target 65000:200 export-extcommunity  
vpn-target 65000:100 import-extcommunity  
vpn-target 65000:200 import-extcommunity  
vpn-target 65000:300 import-extcommunity  
#  
mpls lsr-id 10.8.8.9  
#  
mpls  
#  
mpls ldp  
#  
isis 64  
network-entity 49.0091.0100.0800.8009.00  
#  
interface GigabitEthernet1/0/0  
description toP2GE1/0/2  
undo shutdown  
ip address 10.1.9.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet2/0/0  
description toPE1GE2/0/0  
undo shutdown  
ip address 10.1.16.2 255.255.255.252  
mpls  
mpls ldp  
isis enable 64  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
#
```

```
interface GigabitEthernet3/0/0.10
ip binding vpn-instance NGN_Media
vlan-type dot1q 10
ip address 10.21.1.13 255.255.255.252
#
interface GigabitEthernet3/0/0.11
ip binding vpn-instance NGN_Signaling
vlan-type dot1q 11
ip address 10.21.1.17 255.255.255.252
#
interface GigabitEthernet3/0/0.12
ip binding vpn-instance NGN_Other
vlan-type dot1q 12
ip address 10.21.1.21 255.255.255.252
#
interface LoopBack0
ip address 10.8.8.9 255.255.255.255
isis enable 64
#
bgp 65000
peer 10.11.11.9 as-number 65000
peer 10.11.11.9 connect-interface LoopBack0
#
ipv4-family unicast
undo synchronization
undo peer 10.11.11.9 enable
#
ipv4-family vpng4
policy vpn-target
peer 10.11.11.9 enable
peer 10.11.11.9 route-policy local_pre import
peer 10.11.11.9 route-policy comm export
peer 10.11.11.9 advertise-community
#
ipv4-family vpn-instance NGN_Media
aggregate 10.21.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Other
aggregate 10.21.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Signaling
aggregate 10.21.1.0 255.255.255.0 detail-suppressed
import-route direct
#
route-policy comm permit node 10
apply community 65000:200
#
ip community-filter 1 permit 65000:200
#
route-policy local_pre permit node 10
if-match community-filter 1
apply local-preference 200
#
return
```

- PE3 configuration file

```
#
sysname PE3
#
ip vpn-instance NGN_Media
route-distinguisher 65000:10000811
apply-label per-instance
vpn-target 65000:100 export-extcommunity
vpn-target 65000:100 import-extcommunity
vpn-target 65000:200 import-extcommunity
vpn-target 65000:300 import-extcommunity
ip vpn-instance NGN_Other
route-distinguisher 65000:30000811
```

```
apply-label per-instance
vpn-target 65000:300 export-extcommunity
vpn-target 65000:100 import-extcommunity
vpn-target 65000:200 import-extcommunity
vpn-target 65000:300 import-extcommunity
ip vpn-instance NGN_Signaling
route-distinguisher 65000:20000811
apply-label per-instance
vpn-target 65000:200 export-extcommunity
vpn-target 65000:100 import-extcommunity
vpn-target 65000:200 import-extcommunity
vpn-target 65000:300 import-extcommunity
#
mpls lsr-id 10.9.9.9
#
mpls
#
mpls ldp
#
isis 64
network-entity 49.0091.0100.0900.9009.00
#
interface GigabitEthernet1/0/0
description toP3GE1/0/1
undo shutdown
ip address 10.1.12.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet2/0/0
description toPE4GE2/0/0
undo shutdown
ip address 10.1.17.1 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
undo shutdown
#
interface GigabitEthernet3/0/0.10
ip binding vpn-instance NGN_Media
vlan-type dot1q 10
ip address 10.22.1.73 255.255.255.252
#
interface GigabitEthernet3/0/0.11
ip binding vpn-instance NGN_Signaling
vlan-type dot1q 11
ip address 10.22.1.77 255.255.255.252
#
interface GigabitEthernet3/0/0.12
ip binding vpn-instance NGN_Other
vlan-type dot1q 12
ip address 10.22.1.81 255.255.255.252
#
interface LoopBack0
ip address 10.9.9.9 255.255.255.255
isis enable 64
#
bgp 65000
peer 10.11.11.9 as-number 65000
peer 10.11.11.9 connect-interface LoopBack0
#
ipv4-family unicast
undo synchronization
undo peer 10.11.11.9 enable
#
ipv4-family vpngv4
```

```
policy vpn-target
peer 10.11.11.9 enable
peer 10.11.11.9 route-policy local_pre import
peer 10.11.11.9 route-policy comm export
peer 10.11.11.9 advertise-community
#
ipv4-family vpn-instance NGN_Media
aggregate 10.22.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Other
aggregate 10.22.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Signaling
aggregate 10.22.1.0 255.255.255.0 detail-suppressed
import-route direct
#
route-policy comm permit node 10
apply community 65000:100
#
ip community-filter 1 permit 65000:100
#
route-policy local_pre permit node 10
if-match community-filter 1
apply local-preference 200
#
route-policy local_pre permit node 20
#
return
```

- PE4 configuration file

```
#
sysname PE4
#
ip vpn-instance NGN_Media
route-distinguisher 65000:10000712
apply-label per-instance
vpn-target 65000:100 export-extcommunity
vpn-target 65000:100 import-extcommunity
vpn-target 65000:200 import-extcommunity
vpn-target 65000:300 import-extcommunity
#
ip vpn-instance NGN_Other
route-distinguisher 65000:30000712
apply-label per-instance
vpn-target 65000:300 export-extcommunity
vpn-target 65000:100 import-extcommunity
vpn-target 65000:200 import-extcommunity
vpn-target 65000:300 import-extcommunity
#
ip vpn-instance NGN_Signaling
route-distinguisher 65000:20000712
apply-label per-instance
vpn-target 65000:200 export-extcommunity
vpn-target 65000:100 import-extcommunity
vpn-target 65000:200 import-extcommunity
vpn-target 65000:300 import-extcommunity
#
mpls lsr-id 10.10.10.9
#
mpls
#
mpls ldp
#
isis 64
network-entity 49.0091.0100.1001.0009.00
#
interface GigabitEthernet1/0/0
description toP4GE1/0/1
```

```
undo shutdown
ip address 10.1.14.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet2/0/0
description toPE3GE2/0/0
undo shutdown
ip address 10.1.17.2 255.255.255.252
mpls
mpls ldp
isis enable 64
#
interface GigabitEthernet3/0/0
undo shutdown
#
interface GigabitEthernet3/0/0.10
ip binding vpn-instance NGN_Media
vlan-type dot1q 10
ip address 10.22.1.13 255.255.255.252
#
interface GigabitEthernet3/0/0.11
ip binding vpn-instance NGN_Signaling
vlan-type dot1q 11
ip address 10.22.1.17 255.255.255.252
#
interface GigabitEthernet3/0/0.12
ip binding vpn-instance NGN_Other
vlan-type dot1q 12
ip address 10.22.1.21 255.255.255.252
#
interface LoopBack0
ip address 10.10.10.9 255.255.255.255
isis enable 64
#
bgp 65000
peer 10.11.11.9 as-number 65000
peer 10.11.11.9 connect-interface LoopBack0
#
ipv4-family unicast
undo synchronization
undo peer 10.11.11.9 enable
#
ipv4-family vpng4
policy vpn-target
peer 10.11.11.9 enable
peer 10.11.11.9 route-policy local_pre import
peer 10.11.11.9 route-policy comm export
peer 10.11.11.9 advertise-community
#
ipv4-family vpn-instance NGN_Media
aggregate 10.22.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Other
aggregate 10.22.1.0 255.255.255.0 detail-suppressed
import-route direct
#
ipv4-family vpn-instance NGN_Signaling
aggregate 10.22.1.0 255.255.255.0 detail-suppressed
import-route direct
#
route-policy comm permit node 10
apply community 65000:200
#
ip community-filter 1 permit 65000:200
#
route-policy local_pre permit node 10
```

```
if-match community-filter 1
apply local-preference 200
#
return
```

- RR configuration file

```
#  
sysname RR  
#  
isis 64  
network-entity 49.0091.0100.1101.1009.00  
#  
interface GigabitEthernet1/0/0  
description toP1GE3/0/0  
undo shutdown  
ip address 10.1.3.2 255.255.255.252  
isis enable 64  
#  
interface GigabitEthernet2/0/0  
description toP2GE2/0/0  
undo shutdown  
ip address 10.1.8.2 255.255.255.252  
isis enable 64  
#  
interface LoopBack0  
ip address 10.11.11.9 255.255.255.255  
isis enable 64  
#  
bgp 65000  
group client internal  
peer client connect-interface LoopBack0  
peer 10.7.7.9 as-number 65000  
peer 10.8.8.9 as-number 65000  
peer 10.9.9.9 as-number 65000  
peer 10.10.10.9 as-number 65000  
#  
ipv4-family unicast  
undo synchronization  
undo peer client enable  
undo peer 10.7.7.9 enable  
undo peer 10.8.8.9 enable  
undo peer 10.9.9.9 enable  
undo peer 10.10.10.9 enable  
#  
ipv4-family vpnv4  
undo policy vpn-target  
peer client enable  
peer client reflect-client  
peer client advertise-community  
peer 10.7.7.9 enable  
peer 10.7.7.9 group client  
peer 10.8.8.9 enable  
peer 10.8.8.9 group client  
peer 10.9.9.9 enable  
peer 10.9.9.9 group client  
peer 10.10.10.9 enable  
peer 10.10.10.9 group client  
#  
return
```

Example for Configuring a Basic or Advanced VPN-Target Extended Community Filter

Basic or advanced VPN-target extended community filters can be configured to filter VPN or VPNV4 routes.

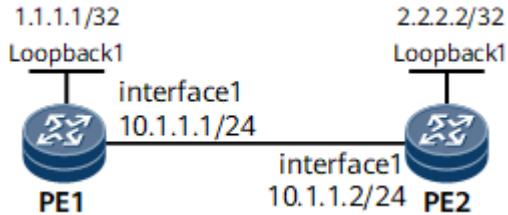
Networking Requirements

In [Figure 1-472](#), an MP-IBGP peer relationship is established between PE1 and PE2. PE2 receives two BGP VPNv4 routes (3.3.3.3/32 and 4.4.4.4/32) from PE1. It is required that a basic or advanced VPN-target extended community filter be configured to filter out the VPNv4 route 4.4.4.4/32 received by PE2.

Figure 1-472 Networking for configuring a basic or advanced VPN-target extended community filter

 NOTE

Interface 1 in this example represents GE 1/0/0.



Configuration Roadmap

The configuration roadmap is as follows:

1. Configure an IP address for each involved interface.
2. Configure basic MPLS and MPLS LDP, and set up MPLS LSPs.
3. Configure VPN instance IPv4 address families on PEs.
4. Establish an MP-IBGP peer relationship between PEs.
5. Configure static routes and import them to VPN instances on PE1.
6. Configure a basic or advanced VPN-target extended community filter on PE2.
7. Configure a route-policy on PE2.
8. Apply the route-policy on PE2 to the routes to be received from PE1.

Data Preparation

To complete the configuration, you need the following data:

- VPN instance names: **vpna**, **vpnb**, and **vpnc**
- ID (1) of a basic VPN-target extended community filter or name (**test**) of an advanced VPN-target extended community filter

Procedure

Step 1 Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic MPLS and MPLS LDP, and set up MPLS LSPs.

Configure PE1.

```
[~PE1] mpls lsr-id 1.1.1.1
[*PE1] mpls
[*PE1-mpls] commit
[~PE1-mpls] quit
[~PE1] mpls ldp
[*PE1-mpls-ldp] commit
[~PE1-mpls-ldp] quit
[~PE1] interface gigabitethernet 1/0/0
[~PE1-GigabitEthernet1/0/0] mpls
[*PE1-GigabitEthernet1/0/0] mpls ldp
[*PE1-GigabitEthernet1/0/0] commit
[~PE1-GigabitEthernet1/0/0] quit
```

Configure PE2.

```
[~PE2] mpls lsr-id 2.2.2.2
[*PE2] mpls
[*PE2-mpls] commit
[~PE2-mpls] quit
[~PE2] mpls ldp
[*PE2-mpls-ldp] commit
[~PE2-mpls-ldp] quit
[~PE2] interface gigabitethernet 1/0/0
[~PE2-GigabitEthernet1/0/0] mpls
[*PE2-GigabitEthernet1/0/0] mpls ldp
[*PE2-GigabitEthernet1/0/0] commit
[~PE2-GigabitEthernet1/0/0] quit
```

Step 3 Configure VPN instance IPv4 address families on PEs.

Configure PE1.

```
[~PE1] ip vpn-instance vpna
[*PE1-vpn-instance-vpna] ipv4-family
[*PE1-vpn-instance-vpna-af-ipv4] route-distinguisher 1:100
[*PE1-vpn-instance-vpna-af-ipv4] vpn-target 1:100 both
[*PE1-vpn-instance-vpna-af-ipv4] quit
[*PE1-vpn-instance-vpna] quit
[*PE1] ip vpn-instance vpngb
[*PE1-vpn-instance-vpngb] ipv4-family
[*PE1-vpn-instance-vpngb-af-ipv4] route-distinguisher 2:100
[*PE1-vpn-instance-vpngb-af-ipv4] vpn-target 2:100 both
[*PE1-vpn-instance-vpngb-af-ipv4] quit
[*PE1-vpn-instance-vpngb] quit
[*PE1] commit
```

Configure PE2.

```
[~PE2] ip vpn-instance vpnc
[*PE2-vpn-instance-vpnc] ipv4-family
[*PE2-vpn-instance-vpnc-af-ipv4] route-distinguisher 1:100
[*PE2-vpn-instance-vpnc-af-ipv4] vpn-target 3:100 export-extcommunity
[*PE2-vpn-instance-vpnc-af-ipv4] vpn-target 1:100 import-extcommunity
[*PE2-vpn-instance-vpnc-af-ipv4] vpn-target 2:100 import-extcommunity
[*PE2-vpn-instance-vpnc-af-ipv4] quit
[*PE2-vpn-instance-vpnc] quit
[*PE2] commit
```

Step 4 Establish an MP-IBGP peer relationship between PEs.

Configure PE1.

```
[~PE1] bgp 100
[*PE1-bgp] peer 2.2.2.2 as-number 100
[*PE1-bgp] peer 2.2.2.2 connect-interface loopback 1
[*PE1-bgp] ipv4-family vpnv4
[*PE1-bgp-af-vpnv4] peer 2.2.2.2 enable
[*PE1-bgp-af-vpnv4] commit
[~PE1-bgp-af-vpnv4] quit
[~PE1-bgp] quit
```

Configure PE2.

```
[~PE2] bgp 100
[*PE2-bgp] peer 1.1.1.1 as-number 100
[*PE2-bgp] peer 1.1.1.1 connect-interface loopback 1
[*PE2-bgp] ipv4-family vpnv4
[*PE2-bgp-af-vpnv4] peer 1.1.1.1 enable
[*PE2-bgp-af-vpnv4] commit
[*PE2-bgp-af-vpnv4] quit
[~PE2-bgp] quit
```

Step 5 Configure static routes and import them to VPN instances on PE1.

```
[~PE1] ip route-static vpn-instance vpna 3.3.3.3 32 NULL0
[*PE1] ip route-static vpn-instance vpnb 4.4.4.4 32 NULL0
[*PE1] commit
[~PE1] bgp 100
[*PE1-bgp] ipv4-family vpn-instance vpna
[*PE1-bgp-vpna] import-route static
[*PE1-bgp-vpna] quit
[*PE1-bgp] ipv4-family vpn-instance vpnb
[*PE1-bgp-vpnb] import-route static
[*PE1-bgp-vpnb] quit
[*PE1-bgp] quit
[*PE1] commit
```

Run the **display bgp vpng4 all routing-table** command on PE2 to check information about BGP VPGN4 routes. The command output shows that two routes 3.3.3.3/32 and 4.4.4.4/32 exist in **vpnc**.

```
[~PE2] display bgp vpng4 all routing-table 3.3.3.3

BGP local router ID : 10.1.1.2
Local AS number : 100

Total routes of Route Distinguisher(1:100): 1
BGP routing table entry information of 3.3.3.3/32:
Label information (Received/Applied): 32905/NULL
From: 1.1.1.1 (1.1.1.1)
Route Duration: 0d00h06m19s
Relay IP Nexthop: 10.1.1.1
Relay IP Out-Interface: GigabitEthernet1/0/0
Relay Tunnel Out-Interface: GigabitEthernet1/0/0
Original nexthop: 1.1.1.1
Qos information : 0x0
Ext-Community: RT <1 : 100>
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Not advertised to any peer yet

VPN-Instance vpnc, Router ID 10.1.1.2:
```

```
Total Number of Routes: 1
BGP routing table entry information of 3.3.3.3/32:
Route Distinguisher: 1:100
Remote-Cross route
Label information (Received/Applied): 32905/NULL
From: 1.1.1.1 (1.1.1.1)
Route Duration: 0d00h06m19s
Relay Tunnel Out-Interface: GigabitEthernet1/0/0
Original nexthop: 1.1.1.1
Qos information : 0x0
Ext-Community: RT <1 : 100>
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Not advertised to any peer yet
```

```
[~PE2] display bgp vpng4 all routing-table 4.4.4.4
```

```
BGP local router ID : 10.1.1.2
Local AS number : 100
```

```
Total routes of Route Distinguisher(2:100): 1
```

```
BGP routing table entry information of 4.4.4.4/32:  
Label information (Received/Applied): 32906/NULL  
From: 1.1.1.1 (1.1.1.1)  
Route Duration: 0d00h06m24s  
Relay IP Nexthop: 10.1.1.1  
Relay IP Out-Interface: GigabitEthernet1/0/0  
Relay Tunnel Out-Interface: GigabitEthernet1/0/0  
Original nexthop: 1.1.1.1  
Qos information : 0x0  
Ext-Community: RT <2 : 100>  
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255  
Not advertised to any peer yet  
  
VPN-Instance vpnc, Router ID 10.1.1.2:  
  
Total Number of Routes: 1  
BGP routing table entry information of 4.4.4.4/32:  
Route Distinguisher: 2:100  
Remote-Cross route  
Label information (Received/Applied): 32906/NULL  
From: 1.1.1.1 (1.1.1.1)  
Route Duration: 0d00h06m24s  
Relay Tunnel Out-Interface: GigabitEthernet1/0/0  
Original nexthop: 1.1.1.1  
Qos information : 0x0  
Ext-Community: RT <2 : 100>  
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255  
Not advertised to any peer yet
```

Step 6 Configure a basic or advanced VPN-target extended community filter on PE2.

Configure a basic VPN-target extended community filter.

```
[~PE2] ip extcommunity-filter 1 index 10 permit rt 1:100  
[*PE2] commit
```

Configure an advanced VPN-target extended community filter.

```
[~PE2] ip extcommunity-filter advanced test index 10 permit ^1:100$  
[*PE2] commit
```

Step 7 Configure a route-policy on PE2.

In the case of a basic VPN-target extended community filter:

```
[~PE2] route-policy test permit node 10  
[*PE2-route-policy] if-match extcommunity-filter 1  
[*PE2-route-policy] quit  
[*PE2] route-policy test deny node 20  
[*PE2] commit
```

In the case of an advanced VPN-target extended community filter:

```
[~PE2] route-policy test permit node 10  
[*PE2-route-policy] if-match extcommunity-filter test  
[*PE2-route-policy] quit  
[*PE2] route-policy test deny node 20  
[*PE2] commit
```

Step 8 Apply the route-policy on PE2 to the routes to be received from PE1.

```
[~PE2] bgp 100  
[~PE2-bgp] ipv4-family vpnv4  
[*PE2-bgp-af-vpnv4] peer 1.1.1.1 route-policy test import  
[*PE2-bgp-af-vpnv4] quit  
[*PE2-bgp] quit  
[*PE2] commit
```

Run the **display bgp vpng4 all routing-table** command on PE2 to check information about BGP VPGv4 routes. The command output shows that route 4.4.4.4/32 has been filtered out.

```
[~PE2] display bgp vpng4 all routing-table 3.3.3.3
```

```
BGP local router ID : 10.1.1.2
Local AS number : 100

Total routes of Route Distinguisher(1:100): 1
BGP routing table entry information of 3.3.3.3/32:
Label information (Received/Applied): 32905/NULL
From: 1.1.1.1 (1.1.1.1)
Route Duration: 0d00h05m41s
Relay IP Nexthop: 10.1.1.1
Relay IP Out-Interface: GigabitEthernet1/0/0
Relay Tunnel Out-Interface: GigabitEthernet1/0/0
Original nexthop: 1.1.1.1
Qos information : 0x0
Ext-Community: RT <1 : 100>
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Not advertised to any peer yet
```

```
VPN-Instance vpnc, Router ID 10.1.1.2:
```

```
Total Number of Routes: 1
BGP routing table entry information of 3.3.3.3/32:
Route Distinguisher: 1:100
Remote-Cross route
Label information (Received/Applied): 32905/NULL
From: 1.1.1.1 (1.1.1.1)
Route Duration: 0d00h37m42s
Relay Tunnel Out-Interface: GigabitEthernet1/0/0
Original nexthop: 1.1.1.1
Qos information : 0x0
Ext-Community: RT <1 : 100>
AS-path Nil, origin incomplete, MED 0, localpref 100, pref-val 0, valid, internal, best, select, pre 255
Not advertised to any peer yet
```

```
[~PE2] display bgp vpng4 all routing-table 4.4.4.4
Info: The network does not exist.
```

----End

Configuration Files

- PE1 configuration file

```
#
sysname PE1
#
ip vpn-instance vpna
 ipv4-family
  route-distinguisher 1:100
  vpn-target 1:100 export-extcommunity
  vpn-target 1:100 import-extcommunity
#
ip vpn-instance vpnb
 ipv4-family
  route-distinguisher 2:100
  vpn-target 2:100 export-extcommunity
  vpn-target 2:100 import-extcommunity
#
mpls lsr-id 1.1.1.1
#
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
```

```
undo shutdown
ip address 10.1.1.1 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 1.1.1.1 255.255.255.255
#
bgp 100
peer 2.2.2.2 as-number 100
peer 2.2.2.2 connect-interface LoopBack1
#
ipv4-family unicast
undo synchronization
peer 2.2.2.2 enable
#
ipv4-family vpng4
policy vpn-target
peer 2.2.2.2 enable
#
ipv4-family vpn-instance vpna
import-route static
#
ipv4-family vpn-instance vpnb
import-route static
#
ip route-static 2.2.2.2 255.255.255.255 GigabitEthernet1/0/0 10.1.1.2
ip route-static vpn-instance vpna 3.3.3.3 255.255.255.255 NULL0
ip route-static vpn-instance vpnb 4.4.4.4 255.255.255.255 NULL0
#
return
```

- PE2 configuration file

```
#
sysname PE2
#
ip vpn-instance vpnc
ipv4-family
route-distinguisher 1:100
vpn-target 3:100 export-extcommunity
vpn-target 1:100 import-extcommunity
vpn-target 2:100 import-extcommunity
#
mpls lsr-id 2.2.2.2
#
mpls
#
mpls ldp
#
interface GigabitEthernet1/0/0
undo shutdown
ip address 10.1.1.2 255.255.255.0
mpls
mpls ldp
#
interface LoopBack1
ip address 2.2.2.2 255.255.255.255
#
bgp 100
peer 1.1.1.1 as-number 100
peer 1.1.1.1 connect-interface LoopBack1
#
ipv4-family unicast
undo synchronization
peer 1.1.1.1 enable
#
ipv4-family vpng4
policy vpn-target
peer 1.1.1.1 enable
peer 1.1.1.1 route-policy test import
```

```
#  
ip extcommunity-filter 1 index 10 permit rt 1:100  
#  
route-policy test permit node 10  
if-match extcommunity-filter 1  
#  
route-policy test deny node 20  
#  
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet1/0/0 10.1.1.1  
#  
return
```

1.1.12 XPL Configuration

1.1.12.1 XPL Description

1.1.12.1.1 Overview of XPL

Definition

Extended routing-policy language (XPL) is a language used to filter routes and modify route attributes. By modifying route attributes (including reachability), XPL changes the path through which network traffic passes. XPL provides the same functions as routing policies do and can meet different customer requirements.

[Table 1-179](#) compares XPL and routing policies.

Table 1-179 Comparison between XPL and routing policies

Item	Key Functions	Editing Method	Filtering Method	User Experience
XPL	Filters routes and modifies route attributes.	Line-by-line or paragraph-by-paragraph editing	Uses sets or single elements to filter routes.	Policies can be flexibly configured and modified.
Routing policies	Filter routes and modify route attributes.	Line-by-line editing	Use filters or single elements to filter routes.	Users must follow strict command configuration rules.



For details about routing policies, see "Routing Policies" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description — IP Routing*.

Purpose

When advertising, receiving, or importing routes, the router can use XPL based on actual networking requirements to filter routes and modify route attributes. XPL serves the following purposes:

- Controls route advertisement.
Only matched routes are advertised.
- Controls route acceptance.
Only necessary and valid routes are accepted, which reduces the routing table size and improves network security.
- Filters and controls imported routes.
A routing protocol may import routes discovered by other routing protocols. XPL ensures that only the routes that meet certain conditions are imported and route attributes of the imported routes are modified to meet the requirements of the protocol.
- Modifies route attributes.
Attributes of the routes that match the specified route-filter can be modified as required.
- Flexibly updates the policies for advertising and accepting routes
XPL supports the paragraph-by-paragraph editing mode. Configurations can be performed in the paragraph editing UI and are committed together after the configurations of a paragraph are complete. Therefore, policy modification and update are relatively flexible.

Benefits

XPL offers the following benefits:

- Saves system resources by controlling the routing table size.
- Improves network security by controlling route advertisement and acceptance.
- Improves network performance by modifying route attributes for effective traffic planning.
- Simplifies routing policy configurations.

1.1.12.1.2 XPL Fundamentals

Understanding XPL

XPL Implementation

XPL implementation involves the following two steps:

1. Define rules: The rules contain the characteristics of the routing information to be filtered. Different attributes in the routing information can be used as the matching basis, such as the destination address and the address of the router that advertises the routing information. The rules are usually defined through **route-filters**.
2. Apply rules: Apply the matching rules to route advertisement, acceptance, and import.

Sets

A set is a group of data that XPL uses as matching rules. XPL sets are classified as global variable sets or route attribute sets.

- Global variable set: A global variable set is a group of frequently used values that are defined as global variables. Global variables are variables that can be referenced by all route-filters on a device. To enable a route-filter to reference a global variable, enter \$+global variable name, for example, **\$glovar1**. The global variables on a device must be unique. A new global variable will override an existing global variable with the same name.
- Route attribute set: A route attribute set is a group of data concerning a route attribute. If the routes to be filtered have the same or similar route attribute, for example, they are destined for the same network segment or originate from the same AS, you can configure a route attribute set for the routes as matching rules. Route attribute sets can be used in route-filters to form filtering rules. **Table 1-180** describes the application scopes and matching items of different route attribute sets.

NOTE

Sets do not have the permit or deny function as routing policies do. Instead, sets are only groups of data used as matching rules, and the actions to be applied are specified in route-filters.

Table 1-180 Route attribute sets

Name	Application Scope	Matching Attribute
IPv4 prefix set	Routes of all dynamic routing protocols	Source, destination, and next hop IP addresses
IPv6 prefix set	Routes of all dynamic routing protocols	Source, destination, and next hop IP addresses
AS_Path set	BGP routes	AS_Path of BGP routes
Community set	BGP routes	Community of BGP routes
Large-Community set	BGP routes	Large-Community of BGP routes
Extended community set	VPN	Route target and Site-of-Origin (SoO) of VPN routes
RD set	VPN	RD of VPN routes

Line-by-Line and Paragraph-by-Paragraph Editing

XPL supports two configuration modes: line-by-line editing and paragraph-by-paragraph editing. Line-by-line editing is a traditional configuration method,

whereas paragraph-by-paragraph editing is an innovative configuration method.
Table 1-181 compares the two modes.

Table 1-181 Line-by-line and paragraph-by-paragraph editing comparison

Item	Applicable to	Main Difference	Help and Error Correction Mechanism
Line-by-line editing	Users who are used to the traditional configuration method or unfamiliar with XPL	Each command is run in a command view, and one command is presented in one line, which is considered a configuration unit. NOTE To modify an existing global variable set, route attribute set, or route-filter using the line-by-line editing mode, you need to enter the specific command view to perform reconfiguration.	You can get the desired command through the type-ahead function. If any configuration error occurs, it is reported after the command is configured.
Paragraph-by-paragraph editing	Users who are familiar with XPL clause configuration and want to simplify the configuration process	The paragraph editing UI functions as a text editor, in which users edit XPL clauses. The XPL clauses are committed after a paragraph of them are configured, and each paragraph is considered a configuration unit.	The type-ahead function is not supported, and complete clauses must be manually entered in the paragraph editing UI. If any configuration error occurs, it is reported after the configurations of the whole paragraph are committed.

Route-Filters

Route-filters are used to filter routes based on sets or a single element and modify route attributes of the routes that match specified rules. A route-filter consists of condition clauses and action clauses. For details about filtering rules, see **Basic Rules of XPL-based Filtering**:

- Condition clause: A condition clause is defined based on a set or single element. The action specified in the action clause is applied only to the routes that match the conditions specified in the condition clause.
- Action clause: An action clause specifies an action to be applied to the routes that match the conditions specified in the condition clause. The system uses action clauses to specify whether routes match the route-filter and set route attributes.

XPL Statements

XPL statements are used to convert filtering rules to global variable sets and route-filters. XPL statements include the remark, set definition, set element, condition clause, action clause, and route-filter with pre-defined variables.

- A remark is an explanation attached to an XPL policy configuration line, beginning with an exclamatory mark (!).

NOTE

If the list content is not empty, the last line (the line right before end-list) cannot be commented out.

```
xpl ip-prefix-list prefix-list1
! prefix-list1 is the name of an ip-prefix-list.
10.0.1.0 24,
10.0.3.0 24 eq 26,
10.0.2.0 24 le 28,
10.0.4.0 24 ge 26 le 30
end-list
```

- A set definition specifies matching rules and begins and ends with apparent clauses.

For example, an IPv4 prefix set begins with **xpl ip-prefix-list** and ends with **end-list**, with a group of IPv4 prefixes in between.

```
xpl ip-prefix-list prefix-list2
10.0.1.0 24,
10.0.3.0 24 eq 26,
10.0.2.0 24 le 28,
10.0.4.0 24 ge 26 le 30
end-list
```

- Set elements include elements such as IP prefixes, AS_Path values, and communities. The elements are separated with commas. Elements in a route-filter must have the same type as the route-filter.

```
xpl ip-prefix-list prefix-list3
10.0.1.0 24,
! element
10.0.3.0 24 eq 26,
! element
10.0.2.0 24 le 28,
! element
10.0.4.0 24 ge 26 le 30
end-list
```

- Condition clauses are used in route-filters and can be used with sets to define matching conditions. Condition clauses consist of **if**, **elseif**, and **else** clauses and may include **eq** (equal to), **ge** (greater than or equal to), **le** (less than or equal to), and **in** (included in) expressions, which can be used in conjunction with the Boolean condition **not**, **and**, or **or**.

in can be followed by a set so that the elements in the set are used as matching rules.

```
xpl route-filter route-filter1
! route-filter1 is the name of Route-Filter
if med eq 20 then
    apply community { 100:1 } additive
endif
end-filter
```

- Action clauses specify the actions to be applied to given routes and include the following clauses:
 - **approve** clause: permits routes and continues to execute the route-filter.

- **refuse** clause: denies routes and stops matching them against the route-filter.
- **finish** clause: permits routes and stops matching them against the route-filter.
- **abort** clause: aborts the route-filter or set modification.
- **apply** clause: permits routes, continues to execute the route-filter, and sets attributes for the routes.
- **call** clause: references other route-filters.
- **break** clause: enables the device to exit the current route-filter but keep executing remaining condition and action clauses of the parent route-filter that references the current route-filter. If the current route-filter is not referenced by any parent route-filter, the device directly exits the current route-filter. The matching result depends on whether an **apply** or **approve** clause has been executed before the matching ends. If an **apply** or **approve** clause has been executed, the route passes the matching. Otherwise, the route is denied.

```
xpl route-filter route-filter2
! Name of Route-Filter
if med eq 10 then
    approve
endif
end-filter
```

- XPL supports route-filters with pre-defined variables. Route-filters with pre-defined variables can be referenced during route-filter configuration through **call** clauses.

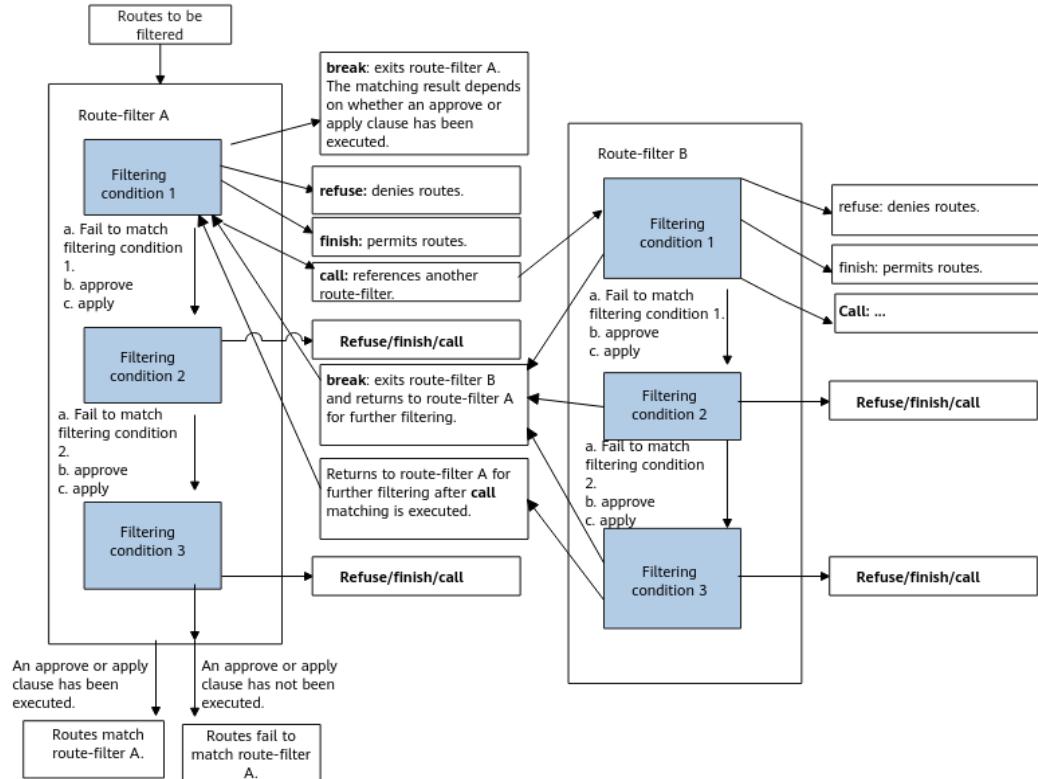
```
xpl route-filter param-route-filter ($mytag)
! Configure a route-filter with a pre-defined variable.
    apply community { 1234:$mytag } additive
end-filter
xpl route-filter origin-10
! Reference the route-filter with the pre-defined variable.
if med eq 20 then
    call route-filter param-route-filter (10)
else
    permit
endif
end-filter
```

Basic Rules of XPL-based Filtering

XPL statements are used to convert filtering rules to global variable sets and route-filters.

Basic Rules of XPL-based Filtering

Figure 1-473 XPL filtering process



As shown in **Figure 1-473**, the routes to be filtered are first matched against filtering conditions 1, 2, and 3 of route-filter A in the configuration sequence. After route matching against each filtering condition, the **refuse**, **finish**, **call**, or **break** clause can be used to stop matching routes against route-filter A. The **refuse** clause determines that routes fail to match route-filter A and stops matching the routes against the route-filter. The **finish** clause determines that the routes match route-filter A and stops matching the routes against the route-filter. The **call** clause exits route-filter A, enters another route-filter for filtering, and then returns to route-filter A for further matching in sequence. The **break** clause exits the current route-filter and returns to the parent route-filter or terminates route-filter-based matching. The **approve** clause allows the routes to match the if condition branch, and the **apply** clause sets route attributes for the routes that match the if condition branch. If the filtering conditions are not met, the routes do not match the if condition branch. If route matching against route-filter A is not stopped, the routes are further matched against other filtering conditions. If the **approve** or **apply** clause has been executed when route matching against route-filter A ends, the routes match route-filter A. Otherwise, the routes do not match route-filter A.

The general filtering rules of XPL route-filters are as follows:

- The **finish** clause stops matching routes against the route-filter and allows the routes to match the route-filter.
 - The **refuse** clause stops matching routes against the route-filter and denies the routes.

- In other cases, routes are further matched against the route-filter in sequence. If the **approve** or **apply** clause has been executed when route matching against the route-filter ends, the routes match the route-filter. Otherwise, the routes do not match the route-filter.
- The **break** clause exits the current route-filter. If a **break** clause is used in the route-filter invoked by the **call** clause, the device exits the current route-filter but goes on with the remaining conditions and actions in the parent route-filter. If no parent route-filter exists, route matching against the current route-filter is terminated. If the **approve** or **apply** clause has been executed when route matching against the route-filter ends, the routes match the route-filter. Otherwise, the routes do not match the route-filter.

Comprehensive XPL Cases

The following describes in detail how to use different XPL clauses at the same time:

```
#  
xpl route-filter sub-filter  
if tag eq 1 then      //This clause is executed if the tag value of the route is 1.  
    apply ip next-hop 7.7.7.7  
elseif tag eq 3 then  //This clause is executed if the tag value of the route is not 1.  
    approve          //This clause is executed if the tag value of the route is 3.  
elseif tag eq 4 then  //This clause is executed if the tag value of the route is not 1 or 3.  
    refuse           //This clause is executed if the tag value of the route is 4. The filtering ends, and the result is deny.  
elseif tag eq 5 then  //This clause is executed if the tag value of the route is not 1, 3, or 4.  
    finish           //This clause is executed if the tag value of the route is 4. The filtering ends, and the result is permit.  
elseif tag eq 6 then  //This clause is executed if the tag value of the route is not 1, 3, 4, or 5.  
    break            //This clause is executed if the tag value of the route is 6. The filtering ends. If the approve or apply clause has been run, the result is permit. Otherwise, the result is deny.  
endif  
if preference eq 1 then  
    apply ip next-hop 9.9.9.9    //This clause is executed if the preference value of the route is 1.  
elseif preference eq 3 then  //This clause is executed if the preference value of the route is not 1.  
    approve          //This clause is executed if the preference value of the route is 3.  
elseif preference eq 4 then  //This clause is executed if the preference value of the route is not 1 or 3.  
    refuse           //This clause is executed if the preference value of the route is 4. The filtering ends, and the result is deny.  
elseif preference eq 5 then  //This clause is executed if the preference value of the route is not 1, 3, or 4.  
    finish           //This clause is executed if the preference value of the route is 5. The filtering ends, and the result is permit.  
elseif preference eq 6 then  //This clause is executed if the preference value of the route is not 1, 3, 4, or 5.  
    break            //This clause is executed if the preference value of the route is 6. The filtering ends. If the approve or apply clause has been run, the result is permit. Otherwise, the result is deny.  
endif  
end-filter  
#  
xpl route-filter father-filter  
if ip route-destination in {2.2.2.1 32} then  
    apply ip next-hop 4.4.4.4    //This clause is executed if the destination address of the route is 2.2.2.1/32.  
elseif ip route-destination in {2.2.2.3 32} then  //This clause is executed if the destination address of the route is not 2.2.2.1/32.  
    approve          //This clause is executed if the destination address of the route is 2.2.2.3/32.  
elseif ip route-destination in {2.2.2.4 32} then  //This clause is executed if the destination address of the route is not 2.2.2.1/32 or 2.2.2.3/32.  
    refuse           //This clause is executed if the destination address of the route is 2.2.2.4/32. The filtering ends, and the result is deny.  
elseif ip route-destination in {2.2.2.5 32} then  //This clause is executed if the destination address of the route is not 2.2.2.1/32, 2.2.2.3/32, or 2.2.2.4/32.  
    finish           //This clause is executed if the destination address of the route is 2.2.2.5/32. The filtering ends, and the result is permit.  
elseif ip route-destination in {2.2.2.6 32} then  //This clause is executed if the destination address of the route is not 2.2.2.1/32, 2.2.2.3/32, 2.2.2.4/32, or 2.2.2.5/32.
```

```
route is not 2.2.2.1/32, 2.2.2.3/32, 2.2.2.4/32, or 2.2.2.5/32.  
call route-filter sub-filter //If the destination address of the route is 2.2.2.6/32, the route-filter sub-filter  
is invoked for filtering. After the filtering using the sub-filter is complete, the device exits the sub-filter and  
resumes filtering in the configuration sequence.  
endif  
if med eq 1 then  
    apply ip next-hop 6.6.6.6 //This clause is executed if the MED value of the route is 1.  
elseif med eq 3 then //This clause is executed if the MED value of the route is not 1.  
    approve //This clause is executed if the MED value of the route is 3.  
elseif med eq 4 then //This clause is executed if the MED value of the route is not 1 or 3.  
    refuse //This clause is executed if the MED value of the route is 4. The filtering ends, and  
the result is deny.  
elseif med eq 5 then //This clause is executed if the MED value of the route is not 1, 3, or 4.  
    finish //This clause is executed if the MED value of the route is 5. The filtering ends, and  
the result is permit.  
endif  
end-filter  
#
```

XPL Action Clauses

Action clauses specify the actions to be applied to given routes.

XPL Action Clauses

Action clauses specify actions in route-filters. [Table 1-182](#) lists XPL action clauses and related examples.

Table 1-182 Example of XPL action clauses

Action Clause	Example
approve	<ol style="list-style-type: none">If no approve or apply clause has been executed when route filtering completes, the matching result is deny by default.If the approve clause is executed, routes are further filtered according to the configuration order. The filtering result is permit.If parallel if condition clauses are used and the approve clause is executed, routes are further filtered according to the configuration order. The filtering result is permit.If the approve clause is executed in the sub-filter invoked through the call clause, routes are further filtered according to the configuration order. The filtering result is permit.

Action Clause	Example
finish	<ol style="list-style-type: none">If the finish clause is executed, the filtering of the route in all filters is terminated regardless of whether the current filter has a parent filter. The apply clause executed previously takes effect for the route, and the filtering result is permit.If the finish clause is executed in parallel if condition clauses, the matching against all filtering conditions is terminated immediately, the executed apply action takes effect, and the filtering result is permit.If the finish clause is executed in nested if condition clauses, the matching against all filtering conditions is terminated immediately, the executed apply action takes effect, and the filtering result is permit.If the finish clause is executed in the sub-filter evoked through the call clause, the matching against all filtering conditions is terminated immediately, the executed apply action takes effect, and the filtering result is permit.
refuse	<ol style="list-style-type: none">If the refuse clause is executed, the matching against all filtering conditions is terminated immediately, the executed apply action does not take effect, and the filtering result is deny.If the refuse clause is executed in parallel if condition clauses, the matching against all filtering conditions is terminated immediately, the executed apply action does not take effect, and the filtering result is deny.If the refuse clause is executed in nested if condition clauses, the matching against all filtering conditions is terminated immediately, the executed apply action does not take effect, and the filtering result is deny.If the refuse clause is executed in the sub-filter evoked through the call clause, the matching against all filtering conditions is terminated immediately, the executed apply action does not take effect, and the filtering result is deny.

Action Clause	Example
break	<ol style="list-style-type: none"> If the break clause is executed in the sub-filter invoked through the call clause, the route filtering ends in the current filter, but not in the parent filter that invokes the current filter, and the remaining conditions and actions in the parent filter continue to be performed. If there is no parent filter and the break clause is executed, the device directly exits the current filter. The filtering result depends on whether the apply or approve clause has been executed. If the apply or approve clause has been executed, the filtering result is permit. Otherwise, the filtering result is deny. If the break clause is executed in parallel if condition clauses, the route filtering ends in the current filter, but not in the parent filter that invokes the current filter, and the remaining conditions and actions in the parent filter continue to be performed. If the break clause is executed in nested if condition clauses, the route filtering ends in the current filter, but not in the parent filter that invokes the current filter, and the remaining conditions and actions in the parent filter continue to be performed.
abort	The abort clause aborts the current configuration.
call	<ol style="list-style-type: none"> If the call clause is executed, the device resumes clause execution in the original filter after completing the filtering in the invoked filter. If the call clause is executed, and the apply clause is executed in the invoked filter, the apply clause takes effect for the route. If multiple parallel call clauses are configured, the device performs the call operations in the configuration sequence. If the call clause is executed in parallel if condition clauses, the device performs the call operations in the configuration sequence. If the call clause is executed in nested if condition clauses, the device performs the call operations in the configuration sequence. If the call clause is executed again in the sub-filter invoked through the call clause, the device performs route filtering in the configuration sequence. When route filtering ends in an invoked filter, the device resumes the filtering in the parent filter.
apply	<ol style="list-style-type: none"> If multiple apply clauses are configured, the device performs apply operations in the configuration sequence. If multiple QPPB configuration groups are referenced by a route-filter, the latest apply group configuration that is executed in the route-filter takes effect. <p>NOTE QPPB is supported only by the NetEngine 8000 X.</p>

The following describes in detail examples corresponding to the implementation rules of XPL action clauses.

approve

- If no **approve** or **apply** clause has been executed when route filtering completes, the matching result is deny by default.

```
#  
xpl route-filter null  
end-filter  
#
```

In this example, the route matching result is deny.

- If the **approve** clause is executed, routes are further filtered according to the configuration order. The filtering result is permit.

```
#  
xpl route-filter approve-filter  
    approve  
end-filter  
#
```

In this example, the **approve** clause is executed, and the filtering result is permit.

- If parallel **if** condition clauses are used and the **approve** clause is executed, routes are further filtered according to the configuration order. The filtering result is permit.

```
#  
xpl route-filter approve-filter  
if ip route-destination in {2.2.2.2 32} then  
    approve  
endif  
if as-path is-none then  
    apply community {65535:1} additive  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32, the **approve** clause is executed, and the filtering result is permit.

If the AS_Path of the route is empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the destination address of the route is 2.2.2.2/32 and the AS_Path attribute is empty, the **approve** clause is executed, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **approve** clause is executed in nested **if** condition clauses, routes are further filtered according to the configuration order. The filtering result is permit.

```
#  
xpl route-filter r2  
if ip route-destination in {1.1.1.1 32} then  
    approve      //If the destination address of the route is 1.1.1.1/32, the approve clause is  
    executed, and the route is further filtered according to the configuration order.  
    if as-path is-none then //If the destination address of the route is 1.1.1.1/32, this if clause is  
    executed to check whether the AS_Path of the route is empty.  
        apply community {65535:1} additive //If the destination address of the route is 1.1.1.1/32 and the  
        AS_Path of the route is empty, this action is performed.  
    endif  
endif  
end-filter  
#
```

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is not empty, the **approve** clause is executed, and the filtering result is permit.

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **approve** clause is executed in the sub-filter invoked through the **call** clause, routes are further filtered according to the configuration order. The filtering result is permit.

```
#  
xpl route-filter r1  
call route-filter r2      //Invoke r2. After completing filtering in r2, the device resumes clause execution in r1 from the next clause in the configuration sequence.  
apply community {65535:1} additive  
end-filter  
#  
xpl route-filter r2  
    approve           //The approve clause is executed, and the device resumes filtering based on the configuration sequence.  
    end-filter  
#
```

In this example, community attribute {65535:1} is added to the route, and the filtering result is permit.

finish

- If the **finish** clause is executed, the matching against all filtering conditions is terminated immediately, the executed **apply** action takes effect, and the filtering result is permit.

```
#  
xpl route-filter finish-filter  
    apply community {65535:1} additive  
    finish  
    apply community {65538:1} additive      //This clause is not executed because the finish clause has been executed.  
    end-filter  
#
```

In this example, community attribute {65535:1} is added to the route, and the filtering result is permit.

- If the **finish** clause is executed in parallel **if** condition clauses, the matching against all filtering conditions is terminated immediately, the executed **apply** action takes effect, and the filtering result is permit.

```
#  
xpl route-filter approve-filter  
if ip route-destination in {2.2.2.2 32} then  
    finish  
endif  
if as-path is-none then  
    apply community {65535:1} additive  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32, the **finish** clause is executed, and the filtering result is permit.

If the destination address of the route is not 2.2.2.2/32 and the AS_Path attribute is empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **finish** clause is executed in nested **if** condition clauses, the matching against all filtering conditions is terminated immediately, the executed **apply** action takes effect, and the filtering result is permit.

```
#  
xpl route-filter r2  
if ip route-destination in {1.1.1.1 32} then  
    if as-path is-none then //If the destination address of the route is 1.1.1.1/32, this if clause is  
        executed to check whether the AS_Path of the route is empty.  
        finish //If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is  
        empty, the finish clause is executed to terminate the matching against all filtering conditions.  
    endif  
    apply community {65535:1} additive //If the destination address of the route is 1.1.1.1/32 and the  
    AS_Path of the route is not empty, this action is performed.  
endif  
end-filter  
#
```

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is empty, the **finish** clause is executed, and the filtering result is permit.

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is not empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **finish** clause is executed in the sub-filter evoked through the **call** clause, the matching against all filtering conditions is terminated immediately, the executed **apply** action takes effect, and the filtering result is permit.

```
#  
xpl route-filter father-filter  
apply community {65535:1} additive  
call route-filter sub-filter  
apply community {65536:1} additive //This clause is not executed because the finish clause has  
been executed in the sub-filter.  
end-filter  
#  
xpl route-filter sub-filter  
finish //The finish clause is executed to immediately terminate the matching against all  
filtering conditions.  
apply community {65538:1} additive //This clause is not executed because the finish clause has  
been executed.  
end-filter  
#
```

In this example, community attribute {65535:1} is added to the route, and the filtering result is permit.

refuse

- If the **refuse** clause is executed, the matching against all filtering conditions is terminated immediately, the executed **apply** action does not take effect, and the filtering result is deny.

```
#  
xpl route-filter finish-filter  
apply community {65535:1} additive  
refuse //The refuse clause is executed to immediately terminate the matching against all  
filtering conditions. The previously executed action does not take effect.  
apply community {65538:1} additive //This clause is not executed because the refuse clause has  
been executed.  
end-filter  
#
```

In this example, the **refuse** clause is executed, and the filtering result is deny.

- If the **refuse** clause is executed in parallel **if** condition clauses, the matching against all filtering conditions is terminated immediately, the executed **apply** action does not take effect, and the filtering result is deny.

```
#  
xpl route-filter approve-filter  
if ip route-destination in {2.2.2.2 32} then
```

```
refuse
endif
if as-path is-none then
    apply community {65535:1} additive
endif
end-filter
#
```

If the destination address of the route is 2.2.2.2/32, the **refuse** clause is executed, and the filtering result is deny.

If the destination address of the route is not 2.2.2.2/32 and the AS_Path attribute is empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **refuse** clause is executed in nested **if** condition clauses, the matching against all filtering conditions is terminated immediately, the executed **apply** action does not take effect, and the filtering result is deny.

```
#  
xpl route-filter r2
if ip route-destination in {1.1.1.1 32} then
    if as-path is-none then //If the destination address of the route is 1.1.1.1/32, this if clause is
        executed to check whether the AS_Path of the route is empty.
        refuse //If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is
        empty, the refuse clause is executed to immediately terminate the matching against all filtering
        conditions.
    endif
    apply community {65535:1} additive //If the destination address of the route is 1.1.1.1/32 and the
    AS_Path of the route is not empty, this action is performed.
    endif
end-filter
#
```

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is empty, the **refuse** clause is executed, and the filtering result is deny.

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is not empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **refuse** clause is executed in the sub-filter evoked through the **call** clause, the matching against all filtering conditions is terminated immediately, the executed **apply** action does not take effect, and the filtering result is deny.

```
#  
xpl route-filter father-filter
call route-filter sub-filter
    apply community {65536:1} additive //This clause is not executed because the refuse clause has
    been executed in the sub-filter.
end-filter
#
xpl route-filter sub-filter
    refuse //The refuse clause is executed to immediately terminate the matching against all
    filtering conditions.
    apply community {65538:1} additive //This clause is not executed because the refuse clause has
    been executed.
end-filter
#
```

In this example, the **refuse** clause is executed, and the filtering result is deny.

break

- If the **break** clause is executed in the sub-filter invoked through the **call** clause, the route filtering ends in the current filter, but not in the parent filter

that invokes the current filter, and the remaining conditions and actions in the parent filter continue to be performed.

```
#  
xpl route-filter r1  
call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
execution in r1 from the next clause in the configuration sequence.  
apply community {65535:1} additive //This action is performed after filtering using r2 is complete.  
end-filter  
#  
xpl route-filter r2  
break      //The break clause is executed to end the filtering in the current filter and return to r1.  
apply community {65536:1} additive //This clause is not executed because the break clause has  
been executed.  
end-filter  
#
```

In this example, the **break** clause is executed, community attribute {65535:1} is added to the route, and the filtering result is permit.

- If there is no parent filter and the **break** clause is executed, the device directly exits the current filter. The filtering result depends on whether the **apply** or **approve** clause has been executed. If the **apply** or **approve** clause has been executed, the filtering result is permit. Otherwise, the filtering result is deny.

```
#  
xpl route-filter r1  
if tag eq 3 then  
    approve  
    break  
    apply community {65536:1} additive //This clause is not executed because the break clause has  
been executed.  
endif  
end-filter  
#
```

If the tag value of the route is 3, the **approve** and **break** clauses are executed, and the filtering result is permit.

If the tag value of the route is not 3, the **break** clause is executed, and the filtering result is deny.

- If the **break** clause is executed in parallel **if** condition clauses, the route filtering ends in the current filter, but not in the parent filter that invokes the current filter, and the remaining conditions and actions in the parent filter continue to be performed.

```
#  
xpl route-filter r1  
call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
execution in r1 from the next clause in the configuration sequence.  
apply community {65535:1} additive //This action is performed after filtering using r2 is complete.  
end-filter  
#  
xpl route-filter r2  
if tag eq 3 then  
    break      //The break clause is executed to end the filtering in the current filter and return to  
route-filter r1.  
endif  
if ip route-destination in {1.1.1.1 32} then //If the tag value of the route is not 3, this if clause is  
executed.  
    apply community {65536:1} additive  
endif  
end-filter  
#
```

If the tag value of the route is 3, the **break** clause is executed, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the tag value of the route is not 3 and the destination address is 1.1.1.1/32, community attributes {65535:1} and {65536:1} are added to the route, and

the filtering result is permit. If the tag value of the route is not 3 and the destination address is not 1.1.1.1/32, community attribute {65535:1} is added to the route, and the filtering result is permit.

- If the **break** clause is executed in nested **if** condition clauses, the route filtering ends in the current filter, but not in the parent filter that invokes the current filter, and the remaining conditions and actions in the parent filter continue to be performed.

```
#  
xpl route-filter r1  
call route-filter r2      //Invoke r2. After completing filtering in r2, the device resumes clause execution in r1 from the next clause in the configuration sequence.  
apply community {65535:1} additive //This action is performed after filtering using r2 is complete.  
end-filter  
#  
xpl route-filter r2  
if tag eq 3 then  
    if ip route-destination in {1.1.1.1 32} then //If the tag value of the route is not 3, this if clause is executed.  
        break      //The break clause is executed to end the filtering in the current filter and return to r1.  
    endif  
    apply community {65536:1} additive  
endif  
end-filter  
#
```

If the tag value of the route is 3 and the destination address is 1.1.1.1/32, the **break** clause is executed, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the tag value of the route is 3 and the destination address is not 1.1.1.1/32, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit. If the tag value of the route is not 3, community attribute {65535:1} is added to the route, and the filtering result is permit.

abort

- The **abort** clause aborts the current configuration.

```
#  
xpl route-filter abort-filter  
if as-path is-none then  
    apply community {65535:1} additive  
endif  
abort  
#
```

If the **abort** clause is executed, the configurations that have not been committed in the current view are discarded and the device returns to the system view, regardless of whether the AS_Path of the route is empty. The **abort** clause is usually used when configurations are incorrect.

call

- If the **call** clause is executed, the device resumes clause execution in the original filter after completing the filtering in the invoked filter.

```
#  
xpl route-filter r1  
call route-filter r2      //Invoke r2. After completing filtering in r2, the device resumes clause execution in r1 from the next clause.  
apply community {65535:1} additive  
end-filter  
#  
xpl route-filter r2  
approve  
end-filter  
#
```

In this example, the **approve** clause is executed, community attribute {65535:1} is added to the route, and the filtering result is permit.

- If the **call** clause is executed, and the **apply** clause is executed in the invoked filter, the **apply** clause takes effect for the route.

```
#  
xpl route-filter r1  
call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
execution in r1 from the next clause.  
apply community {65535:1} additive  
end-filter  
#  
xpl route-filter r2  
apply community {65536:1} additive  
end-filter  
#
```

In this example, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

- If multiple parallel **call** clauses are configured, the device performs the **call** operations in the configuration sequence.

```
#  
xpl route-filter r1  
call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
execution in r1 from the next clause.  
call route-filter r3    //Invoke r3. After completing filtering r3, the device resumes clause  
execution in r1 from the next clause.  
end-filter  
#  
xpl route-filter r2  
if tag eq 3 then  
    apply community {65535:1} additive  
endif  
end-filter  
#  
xpl route-filter r3  
if as-path is-none then  
    apply community {65536:1} additive  
endif  
end-filter  
#
```

If the tag value of the route is 3, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the AS_Path of the route is empty, community attribute {65536:1} is added to the route, and the filtering result is permit.

If the AS_Path of the route is empty and the tag value is 3, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **call** clause is executed in parallel **if** condition clauses, the device performs the **call** operations in the configuration sequence.

```
#  
xpl route-filter r1  
if tag eq 3 then  
    call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
execution in r1 from the next clause.  
endif  
if as-path is-none then  
    call route-filter r3    //Invoke r3. After completing filtering r3, the device resumes clause  
execution in r1 from the next clause.  
endif  
end-filter  
#
```

```
xpl route-filter r2
    apply community {65535:1} additive
end-filter
#
xpl route-filter r3
    apply community {65536:1} additive
end-filter
#
```

If the tag value of the route is 3, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the AS_Path of the route is empty, community attribute {65536:1} is added to the route, and the filtering result is permit.

If the AS_Path of the route is empty and the tag value is 3, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **call** clause is executed in nested **if** condition clauses, the device performs the **call** operations in the configuration sequence.

```
#  
xpl route-filter r1  
if tag eq 3 then  
    if as-path is-none then  
        call route-filter r3    //Invoke r3. After completing filtering r3, the device resumes clause  
        execution in r1 from the next clause.  
    endif  
    call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
    execution in r1 from the next clause.  
endif  
end-filter  
#  
xpl route-filter r2  
    apply community {65535:1} additive
end-filter
#
xpl route-filter r3
    apply community {65536:1} additive
end-filter
#
```

If the tag value of the route is 3 and the AS_Path of the route is not empty, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the tag value of the route is 3 and the AS_Path of the route is empty, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **call** clause is executed again in the sub-filter invoked through the **call** clause, the device performs route filtering in the configuration sequence. When route filtering ends in an invoked filter, the device resumes the filtering in the parent filter.

```
#  
xpl route-filter r1  
call route-filter r2    //Invoke r2. After completing filtering in r2, the device resumes clause  
execution in r1 from the next clause.  
end-filter  
#  
xpl route-filter r2  
call route-filter r3    //Invoke r3. After completing filtering r3, the device resumes clause  
execution in r1 from the next clause.  
    apply community {65535:1} additive
end-filter
```

```
#  
xpl route-filter r3  
if tag eq 3 then  
    apply community {65535:1} additive  
endif  
end-filter  
#
```

If the tag value of the route is 3, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

Otherwise, community attribute {65535:1} is added to the route, and the filtering result is permit.

apply

- If multiple **apply** clauses are configured, the device performs **apply** operations in the configuration sequence.

```
#  
xpl route-filter test001  
apply community {65535:1} additive  
apply community {65536:1} additive  
end-filter  
#
```

In this example, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

- If multiple QPPB configuration groups are referenced by a route-filter, the latest **apply group** configuration that is executed in the route-filter takes effect.

NOTE

QPPB is supported only by the NetEngine 8000 X.

```
#  
xpl route-flow-group rfg1  
apply ip-precedence 6  
end-group  
#  
xpl route-flow-group rfg2  
apply qos-local-id 39  
end-group  
#  
xpl route-filter rf1  
apply group rfg1  
apply group rfg2  
end-filter  
#
```

Common XPL Condition Clauses

Condition clauses are used in route-filters and can be used with sets to define matching conditions. Condition clauses can be **if**, **elseif**, or **else** clauses. [Table 1-183](#) lists common XPL condition clauses and provides examples.

Common XPL Condition Clauses

Table 1-183 Implementation rules of common XPL condition clauses

Condition clause	Example
if	1. When parallel if clauses are configured, the device performs filtering against the condition clauses in the configuration sequence. 2. When nested if clauses are configured, the device performs filtering against the condition clauses based on the logical relationship.
elseif	The elseif clause filters the routes that fail to meet the filtering conditions specified in the previous if or elseif clause of the same if condition branch.
else	The else clause performs the action on all the routes that fail to meet the previous filtering conditions in the same if condition branch.
matches-all	If the matches-all clause is executed, a route matches the route-filter only if it meets all the conditions set in the condition clauses of the route-filter.
The address family type of the route is different from that in the route-filter.	If the prefix type of a route is different from that in the route-filter (for example, the prefix type of a route is IPv4 but the prefix type in the route-filter is IPv6), the route matches the condition clause by default.

The following describes in detail examples corresponding to the implementation rules of common XPL condition clauses.

if

- When parallel **if** clauses are configured, the device performs filtering against the condition clauses in the configuration sequence.

```
#  
xpl route-filter r1  
if ip route-destination in {1.1.1.1 32} then  
    apply community {65535:1} additive  
endif  
if as-path is-none then  
    apply community {65536:1} additive  
endif  
end-filter  
#
```

If the destination address of the route is 1.1.1.1/32, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the AS_Path of the route is empty, community attribute {65536:1} is added to the route, and the filtering result is permit.

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is empty, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

- When nested **if** clauses are configured, the device performs filtering against the condition clauses based on the logical relationship.

```
#  
xpl route-filter r2  
if ip route-destination in {1.1.1.1 32} then  
    if as-path is-none then //If the destination address of the route is 1.1.1.1/32, this if clause is  
        executed to check whether the AS_Path of the route is empty.  
        apply community {65535:1} additive //If the destination address of the route is 1.1.1.1/32 and  
        the AS_Path of the route is empty, this action is performed.  
    endif  
    apply community {65536:1} additive //If the destination address of the route is 1.1.1.1/32, this  
    action is performed.  
endif  
end-filter  
#
```

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is not empty, community attribute {65536:1} is added to the route, and the filtering result is permit.

If the destination address of the route is 1.1.1.1/32 and the AS_Path of the route is empty, community attributes {65535:1} and {65536:1} are added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

elseif

- The **elseif** clause filters the routes that fail to meet the filtering conditions specified in the previous **if** or **elseif** clause of the same **if** condition branch.

```
#  
xpl route-filter if-elseif-filter  
if ip route-destination in {2.2.2.2 32} then  
    apply community {65535:1} additive  
    finish  
elseif apply tag 30 then //If the destination address of the route is not 2.2.2.2 /32, this if clause  
    is executed to check whether the tag value of the route is 30.  
    apply community {65536:1} additive  
    finish  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the destination address of the route is not 2.2.2.2/32 and the tag value of the route is 30, community attribute {65536:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

matches-all

- If the **matches-all** clause is executed, a route matches the route-filter only if it meets all the conditions set in the condition clauses of the route-filter.

```
#  
xpl route-filter regular-filter  
if community matches-all {700, 800} then  
    apply community {65535:1} additive  
    finish  
endif
```

```
end-filter
#
```

If the community value of the route contains both 700 and 800, community attribute {65535:1} is added to the route, and the filtering result is permit.

Otherwise, the filtering result is deny.

else

- The **else** clause performs the action on all the routes that fail to meet the previous filtering conditions in the same **if** condition branch.

```
#  
xpl route-filter if-else-filter  
if ip route-destination in {2.2.2.2 32} then  
    apply community {65535:1} additive  
else  
    apply community {65536:1} additive  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32, community attribute {65535:1} is added to the route, and the filtering result is permit.

If the destination address of the route is not 2.2.2.2/32, community attribute {65536:1} is added to the route, and the filtering result is permit.

The address family type of the route is different from that in the route-filter.

- If the prefix type of a route is different from that in the route-filter (for example, the prefix type of a route is IPv4 but the prefix type in the route-filter is IPv6), the route matches the condition clause by default.

```
#  
bgp 100  
peer 1::1 as-number 100  
#  
ipv6-family unicast  

```

In this example, the peer address (next hop address of a BGP IPv4 public network route) of a BGP IPv6 public network peer is an IPv6 prefix address, and the prefix type of the route to be filtered is different from that in the route-filter. Therefore, the filtering result is permit.

XPL Condition Clause Operators

Condition Clause Operators

In XPL clauses, logical operations such as **not**, **and**, and **or** can be performed between conditions. **Table 1-184** lists condition clause operators and related examples. **Table 1-185** describes the usage rules and forms of different operators.

Table 1-184 Operator examples

Operator	Example
and	A route matches an if clause with two conditions connected with and only when it meets both the two conditions. In this case, the action specified in the if clause is performed.
or	A route matches an if clause with two conditions connected with or if it meets either of the two conditions. In this case, the action specified in the if clause is performed.
not	If a route does not meet the condition following not in an if clause, the route matches the if clause and the action specified in the if clause is performed.
Condition combination	In clauses, and and or combine conditions from left to right, and not combines conditions from right to left. You are advised to add parentheses when configuring complex condition clauses.

Table 1-185 Rules for using operators

Priority	Operator	Name or Meaning	Format	Combination Direction
1	()	Parentheses	(Expression)	From left to right
2	not	Logical NOT	not expression	From right to left
3	and	Logical AND	Expression and expression	From left to right
4	or	Logical OR	Expression or expression	From left to right

NOTE

The combination direction specifies the rule that determines which operator is executed first when several operators have the same priority.

- From left to right: If the priorities are the same, calculation is performed from left to right.
- From right to left: If operators have the same priority, calculation is performed from right to left.

The following describes in detail examples corresponding to the implementation rules of XPL condition clause operators.

and

- A route matches an **if** clause with two conditions connected with **and** only when it meets both the two conditions. In this case, the action specified in the **if** clause is performed.

```
#  
xpl route-filter and-filter  
if ip route-destination in {2.2.2.2 32} and as-path is-none then  
    finish //If the destination address of the route is 2.2.2.2/32 and the AS_Path of the route is  
empty, the finish clause is executed.  
else  
    refuse //If the destination address of the route is not 2.2.2.2/32 or the AS_Path of the route is  
empty, the refuse clause is executed.  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32 and the AS_Path of the route is empty, the filtering result is permit.

If the destination address of the route is not 2.2.2.2/32 or the AS_Path of the route is empty, the filtering result is deny.

or

- A route matches an **if** clause with two conditions connected with **or** if it meets either of the two conditions. In this case, the action specified in the **if** clause is performed.

```
#  
xpl route-filter or-filter  
if ip route-destination in {2.2.2.2 32} or as-path is-none then  
    finish //If the destination address of the route is 2.2.2.2/32 or the AS_Path of the route is empty,  
the finish clause is executed.  
else  
    refuse //If the destination address of the route is not 2.2.2.2/32 and the AS_Path of the route is  
not empty, the refuse clause is executed.  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32 or the AS_Path of the route is empty, the filtering result is permit.

If the destination address of the route is not 2.2.2.2/32 and the AS_Path of the route is not empty, the filtering result is deny.

not

- If a route does not meet the condition following **not** in an **if** clause, the route matches the **if** clause and the action specified in the **if** clause is performed.

```
#  
xpl route-filter not-filter  
if not ip route-destination in {2.2.2.2 32} then  
    finish //If the destination address of the route is not 2.2.2.2 /32, the finish clause is executed.  
else  
    refuse //If the destination address of the route is 2.2.2.2/32, the refuse clause is executed.  
endif  
end-filter  
#
```

If the destination address of the route is not 2.2.2.2/32, the filtering result is permit.

If the destination address of the route is 2.2.2.2/32, the filtering result is deny.

Combination of condition clauses

- In clauses, **and** and **or** combine conditions from left to right, and **not** combine conditions from right to left. You are advised to add parentheses when configuring complex condition clauses.

```
#  
xpl route-filter or-filter  
if ip route-destination in {2.2.2.2 32} and as-path is-none or ip route-destination in {3.3.3.3 32} then  
    finish  
else  
    refuse  
endif  
end-filter  
#
```

If the destination address of the route is 2.2.2.2/32 and the AS_Path of the route is empty, or the destination address of the route is 3.3.3.3/32, the filtering result is permit.

Otherwise, the filtering result is deny.

Matching Rules for XPL Condition Clauses Using Regular Expressions

A regular expression defines a string matching mode. It consists of common characters (such as letters from a to z) and special characters (also called metacharacters).

Matching Rules for Condition Clauses Using Regular Expressions

A regular expression defines a string matching mode. It consists of common characters (such as letters from a to z) and special characters (also called metacharacters). The regular expression functions as a template to match a character pattern with the searched character string. . [Table 1-186](#) lists common XPL condition clauses and provides examples.

NOTE

XPL condition clauses do not support question marks (?) or {x,y}. In the filtering that is based on a community attribute or extended community attribute, the special characters ^ and \$ do not take effect.

Table 1-186 Examples of matching against condition clauses using regular expressions

Matching Attribute	Example
as-path	<ol style="list-style-type: none"> The underscore (_) matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning or end of a regular expression. If ^ is used as the first character of a regular expression, it matches the routes with the specified beginning of the AS_Path attribute line. If \$ is used as the last character of a regular expression, it matches the routes with the specified end of the AS_Path attribute line. If both ^ and \$ are used, both the beginning and end of the AS_Path attribute line are matched. If .* is used in the middle, the conditions at the beginning and end of the line must be met. If both ^ and \$ are used, both the beginning and end of the AS_Path attribute line are matched. If a vertical bar () is used in the middle, the matching is successful when any position at the beginning or end of a line meets the condition. [a-z] matches any character within the range [a-z] specified in the regular expression in the corresponding position.
Community attribute	<ol style="list-style-type: none"> The underscore (_) matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning or end of a regular expression. x y matches x or y. If the match all clause is executed, a route passes the filtering only when it meets all the filtering conditions. [a-z] matches any character within the range specified in the regular expression in the corresponding position. If a community attribute is used by a route-filter, a route matches the route-filter even if the format of the community attribute carried in the route is different from that specified in the route-filter. Well-known community attributes can be used to replace numeric expressions to filter routes.

Matching Attribute	Example
Extended community attribute	<ol style="list-style-type: none"> 1. The underscore (_) matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning or end of a regular expression. 2. xly matches x or y. 3. If the match all clause is executed, a route passes the filtering only when it meets all the filtering conditions. 4. [a-z] matches any character within the range [a-z] specified in the regular expression in the corresponding position. 5. If an extended community attribute is used by a route-filter, the format of the extended community attribute carried in the route must be the same as that specified in the route-filter so that the route matches the route-filter.

The following describes in detail examples corresponding to the implementation rules of XPL condition clauses using regular expressions.

AS_Path attribute-based matching (for details, see [AS_Path](#).)

- The underscore (_) matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning or end of a regular expression.
 - If an underscore (_) is added to a route-filter, only the routes with the specified AS_Path value can match the route-filter.

```
#  
xpl as-path-list as-list  
regular 700_  
end-list  
#  
xpl route-filter regular-filter  
if as-path in as-list then  
  finish  
endif  
end-filter  
#
```

If the AS_Path value of the route is 700, the filtering result is permit. Otherwise, the filtering result is deny.

- If no underscore (_) is added to the route-filter, a route matches the route-filter as long as the AS_Path of the route contains the specified value.

```
#  
xpl community-list comm-list  
regular 700  
end-list  
#  
xpl route-filter regular-filter  
if community matches-any comm-list then  
  finish  
endif  
end-filter  
#
```

If the AS_Path value of a route contains 700, for example, the AS_Path value of the route is 7000 or 70000, the filtering result is permit.

Otherwise, the filtering result is deny.

- If ^ is used as the first character of a regular expression, it matches the routes with the specified beginning of the AS_Path attribute line.

```
#  
xpl as-path-list as-list  
regular ^(700|800)_  
end-list  
#  
xpl route-filter regular-filter  
if as-path in as-list then  
    finish  
endif  
end-filter  
#
```

If the AS_Path attribute line of a route begins with 700 or 800, the filtering result is permit.

Otherwise, the filtering result is deny.

- If \$ is used as the last character of a regular expression, it matches the routes with the specified end of the AS_Path attribute line.

```
#  
xpl as-path-list as-list  
regular _(700|800)$  
end-list  
#  
xpl route-filter regular-filter  
if as-path in as-list then  
    finish  
endif  
end-filter  
#
```

If the AS_Path attribute line of a route ends with 700 or 800, the filtering result is permit.

Otherwise, the filtering result is deny.

- If both ^ and \$ are used, both the beginning and end of the AS_Path attribute line are matched. If .* is used in the middle (indicating that the AS_Path can contain any characters except the first and last characters), the conditions at the beginning and end of the line must be met.

```
#  
xpl as-path-list as-list  
regular ^(500|600)_.*_(700|800)$|(^500|600)_(700|800)$  
end-list  
#  
xpl route-filter regular-filter  
if as-path in as-list then  
    finish  
endif  
end-filter  
#
```

If the AS_Path attribute line of a route starts with 500 or 600 and ends with 700 or 800, the filtering result is permit.

Otherwise, the filtering result is deny.

 NOTE

When only the **regular(^line beginning value a|line beginning value b)_.*_(line end value c|line end value d)\$** expression is used for filtering, the routes whose AS_Paths contain the specified line beginning value and line end value but do not contain other characters cannot pass the filtering. Therefore, to allow all routes that meet the requirements at the line beginning and end to pass the filtering, refer to the regular expression clause in this example.

- If both ^ and \$ are used, both the beginning and end of the AS_Path attribute line are matched. If a vertical bar (|) is used in the middle, the matching is successful when any position at the beginning or end of a line meets the condition.

```
#  
xpl as-path-list as-list  
regular ^(500|600)_\_(700|800)$  
end-list  
#  
xpl route-filter regular-filter  
if as-path in as-list then  
    finish  
endif  
end-filter  
#
```

If the AS_Path attribute line of a route starts with 500 or 600 or ends with 700 or 800, the filtering result is permit.

Otherwise, the filtering result is deny.

- **[a-z]** matches any character within the range **[a-z]** specified in the regular expression in the corresponding position.

```
#  
xpl as-path-list as-list  
regular 60[2-4]_  
end-list  
#  
xpl route-filter regular-filter  
if as-path in as-list then  
    finish  
endif  
end-filter  
#
```

If the AS_Path value of a route is 602, 603, or 604, the filtering result is permit.

Otherwise, the filtering result is deny.

Community attribute-based matching (for details, see [Community Attribute](#).)

- The underscore (_) matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning or end of a regular expression.
 - If an underscore (_) is added to a route-filter, only the routes with the specified community attribute value can match the route-filter.

```
#  
xpl community-list comm-list  
regular 700_  
end-list  
#  
xpl route-filter regular-filter  
if community matches-any comm-list then  
    finish  
endif  
end-filter  
#
```

If the community attribute value of a route is 700, the filtering result is permit.

Otherwise, the filtering result is deny.

- If no underscore (_) is added to the route-filter, a route matches the route-filter as long as the community attribute of the route contains the specified value.

```
#  
xpl community-list comm-list  
regular 700  
end-list  
#  
xpl route-filter regular-filter  
if community matches-any comm-list then  
    finish  
endif  
end-filter  
#
```

If the community attribute value of a route contains 700, for example, the community attribute of the route is 7000 or 70000, the filtering result is permit.

Otherwise, the filtering result is deny.

- **x|y** matches x or y.

```
#  
xpl community-list comm-list  
regular 700_|800_  
end-list  
#  
xpl route-filter regular-filter  
if community matches-any comm-list then  
    finish  
endif  
end-filter  
#
```

If the community attribute value of a route is 700 or 800, the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **match all** clause is executed, a route passes the filtering only when it meets all the filtering conditions.

```
#  
xpl community-list c1  
regular 700_  
regular 800_  
end-list  
#  
xpl route-filter xp1  
if community matches-all c1 then  
    approve  
else  
    refuse  
endif  
end-filter  
#
```

If the community attribute of a route contains both 700 and 800, the filtering result is permit.

Otherwise, the filtering result is deny.

- **[a-z]** matches any character within the range specified in the regular expression in the corresponding position.

```
#  
xpl community-list comm-list
```

```
regular 50[2-4]_
end-list
#
xpl route-filter regular-filter
if community matches-any comm-list then
    finish
endif
end-filter
#
```

If the community attribute value of a route is 502, 503, or 504, the filtering result is permit.

Otherwise, the filtering result is deny.

- If a community attribute is used by a route-filter, a route matches the route-filter even if the format of the community attribute carried in the route is different from that specified in the route-filter.

Decimal integer format

```
#
xpl community-list comm-list
regular 500_
end-list
#
xpl route-filter regular-filter
if community matches-any comm-list then
    finish
endif
end-filter
#
```

Hexadecimal format aa:nn

```
#
xpl community-list comm-list
regular 0:500_
end-list
#
xpl route-filter regular-filter
if community matches-any comm-list then
    finish
endif
end-filter
#
```

No matter which of the preceding formats is used to represent the community attribute in the route-filter, the filtering result is permit as long as the community attribute of the route is 500 or 0:500.

Otherwise, the filtering result is deny.

- Well-known community attributes can be used to replace numeric expressions to filter routes.

```
#
xpl community-list comm-list
regular no
end-list
#
xpl route-filter regular-filter
if community matches-any as-list then
    finish
endif
end-filter
#
```

If the community attribute of a route is no-advertise (65535:65281), no-export (65535:65282), or no-export-subconfed (65535:65283), the filtering result is permit.

Otherwise, the filtering result is deny.

Extended community attribute-based matching

- The underscore (_) matches a sign, such as a comma (,), left brace ({), right brace (}), left parenthesis ((), right parenthesis ()), and space. In addition, it can be used at the beginning or end of a regular expression.

- If an underscore (_) is added to a route-filter, only the routes with the specified extended community attribute value can match the route-filter.

```
#  
xpl extcommunity-list rt rt-list  
regular 1:1_  
end-list  
#  
xpl route-filter regular-filter  
if extcommunity rt matches-any rt-list then  
    finish  
endif  
end-filter  
#
```

If the extended community attribute value of a route is 1:1, the filtering result is permit.

Otherwise, the filtering result is deny.

- If no underscore (_) is added to the route-filter, a route matches the route-filter as long as the extended community attribute of the route contains the specified value.

```
#  
xpl extcommunity-list rt rt-list  
regular 1:1  
end-list  
#  
xpl route-filter regular-filter  
if extcommunity rt matches-any rt-list then  
    finish  
endif  
end-filter  
#
```

If the extended community attribute value of a route contains 1:1, for example, 1:10 or 1:11, the filtering result is permit.

Otherwise, the filtering result is deny.

- x|y** matches x or y.

```
#  
xpl extcommunity-list rt rt-list  
regular 1:1|1:2_  
end-list  
#  
xpl route-filter regular-filter  
if extcommunity rt matches-any rt-list then  
    finish  
endif  
end-filter  
#
```

If the extended community attribute value of a route is 1:1 or 1:2, the filtering result is permit.

Otherwise, the filtering result is deny.

- If the **match all** clause is executed, a route passes the filtering only when it meets all the filtering conditions.

```
#  
xpl extcommunity-list c1  
regular 700_  
regular 800_  
end-list
```

```
#  
xpl route-filter xp1  
if extcommunity matches-all c1 then  
    approve  
else  
    refuse  
endif  
end-filter  
#
```

If the extended community attribute of a route contains both 700 and 800, the filtering result is permit.

Otherwise, the filtering result is deny.

- **[a-z]** matches any character within the range [a-z] specified in the regular expression in the corresponding position.

```
#  
xpl extcommunity-list rt rt-list  
regular 1:[1-2]]_  
end-list  
#  
xpl route-filter regular-filter  
if extcommunity rt matches-any rt-list then  
    finish  
endif  
end-filter  
#
```

If the extended community attribute value of a route is 1:1 or 1:2, the filtering result is permit.

Otherwise, the filtering result is deny.

- If an extended community attribute is used by a route-filter, the format of the extended community attribute carried in the route must be the same as that specified in the route-filter so that the route matches the route-filter.

IPv4 address:2-byte user-defined number

```
#  
xpl extcommunity-list rt rt-list  
regular 192.168.122.15:1_  
end-list  
#  
xpl route-filter regular-filter  
if extcommunity rt matches-any rt-list then  
    finish  
endif  
end-filter  
#
```

4-byte AS number in dotted notation:2-byte user-defined number

```
#  
xpl extcommunity-list rt rt-list  
regular 1.2:3_  
end-list  
#  
xpl route-filter regular-filter  
if extcommunity rt matches-any rt-list then  
    finish  
endif  
end-filter  
#
```

AS number:user-defined number

```
#  
xpl extcommunity-list rt rt-list  
regular 65535:1_  
end-list  
#
```

```
xpl route-filter regular-filter
if extcommunity rt matches-any rt-list then
    finish
endif
end-filter
#
```

The filtering result is permit only when the extended community attribute value of a route contains the same values as those in the corresponding route-filter.

Otherwise, the filtering result is deny.

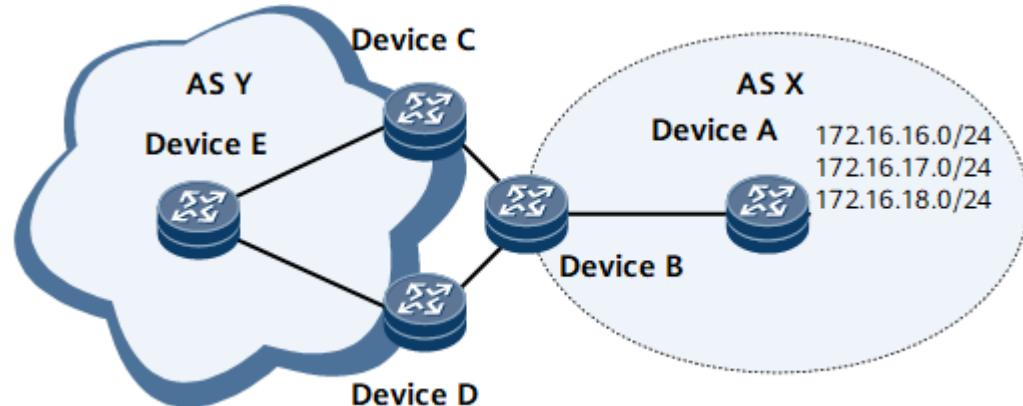
1.1.12.1.3 Application Scenarios for XPL

Using XPL to Filter Routes

On the network shown in [Figure 1-474](#), Device A has routes to 172.16.16.0/24, 172.16.17.0/24, and 172.16.18.0/24. The networking requirements are as follows:

- Device A advertises only route 172.16.17.0/24 to Device B.
- After receiving the route from Device B, Device C forwards it directly to Device E, and Device D increases the MED attribute of the route before forwarding it to Device E to ensure that Device C functions as the egress for the traffic from Device E to 172.16.17.0/24.

[Figure 1-474](#) Route filtering scenario



The preceding requirements can be met using an IPv4 prefix set:

1. Configure an IPv4 prefix set named **ip-prefix1**, which includes only the 172.16.17.0/24 element, on Device A.
2. Configure a route-filter named **route-filter1** on Device A, which permits the route carrying the element in **ip-prefix1** and denies other routes.
3. Configure **route-filter1** as an export policy on Device A so that Device A advertises only route 172.16.17.0/24 to Device B.
4. Configure an IPv4 prefix set named **ip-prefix2**, which includes only 172.16.17.0/24, on Device D.
5. Configure a route-filter named **route-filter2**, which increases the MED value of the route carrying the element in **ip-prefix2**, on Device D.

6. Configure **route-filter2** as an export policy on Device D so that the MED value of the route advertised by Device D is greater than that of the route advertised by Device C, making Device C the egress for the traffic from Device E to 172.16.17.0/24.

1.1.12.2 XPL Configuration

Extended routing-policy language (XPL) is a language used to filter routes and modify route attributes. By modifying route attributes (including reachability), XPL changes the path through which network traffic passes.

1.1.12.2.1 Overview of XPL

This section describes the basic concepts of XPL, comparison between XPL and routing policies, and two XPL configuration modes.

Definition

Extended routing-policy language (XPL) is a language used to filter routes and modify route attributes. By modifying route attributes (including reachability), XPL changes the path through which network traffic passes. XPL provides the same functions as routing policies do and can meet different customer requirements.

Table 1-187 compares XPL and routing policies.

Table 1-187 Comparison between XPL and routing policies

Item	Key Functions	Editing Method	Filtering Method	User Experience
XPL	Filters routes and modifies route attributes.	Line-by-line or paragraph-by-paragraph editing	Uses sets or single elements to filter routes.	Policies can be flexibly configured and modified.
Routing policies	Filter routes and modify route attributes.	Line-by-line editing	Use filters or single elements to filter routes.	Users must follow strict command configuration rules.



For details about routing policies, see "Routing Policies" in *HUAWEI NetEngine 8100 X/NetEngine 8000 X/NetEngine 8000E X series Router Feature Description — IP Routing*.

Line-by-Line and Paragraph-by-Paragraph Editing

XPL supports two configuration modes: line-by-line editing and paragraph-by-paragraph editing.

The line editing mode is a traditional configuration mode on the device and is not described in this section. For details, see the set and route-filter configuration procedures in the following sections. During line-by-line editing of a set or route-filter, you can run the **abort** command to cancel the configurations that have not been committed in the current view and return to the system view or run the **display this candidate** command to check the configurations that have not been committed in the current view.

The paragraph editing mode is a new attempt to configure XPL. [Table 1-188](#) compares the two modes. For details about paragraph editing, see [1.1.12.2.3 Introduction to XPL Paragraph-by-Paragraph Editing](#).

For details about XPL paragraph editing clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Table 1-188 Line-by-line and paragraph-by-paragraph editing comparison

Item	Applicable to	Main Difference	Help and Error Correction Mechanism
Line-by-line editing	Users who are used to the traditional configuration method or unfamiliar with XPL	Each command is run in a command view, and one command is presented in one line, which is considered a configuration unit. NOTE To modify an existing global variable set, route attribute set, or route-filter using the line-by-line editing mode, you need to enter the specific command view to perform reconfiguration.	You can get the desired command through the type-ahead function. If any configuration error occurs, it is reported after the command is configured.
Paragraph-by-paragraph editing	Users who are familiar with XPL clause configuration and want to simplify the configuration process	Configurations are performed in the paragraph editing UI and are committed together after the configurations of a paragraph are complete. Each paragraph is considered a configuration unit.	The type-ahead function is not supported, and complete clauses must be manually entered in the paragraph editing UI. If any configuration error occurs, it is reported after the configurations of the whole paragraph are committed.

Purpose

When advertising, receiving, or importing routes, the router can use XPL based on actual networking requirements to filter routes and modify route attributes. XPL serves the following purposes:

- Controls route advertisement.
Only matched routes are advertised.
- Controls route acceptance.
Only necessary and valid routes are accepted, which reduces the routing table size and improves network security.
- Filters and controls imported routes.
A routing protocol may import routes discovered by other routing protocols. XPL ensures that only the routes that meet certain conditions are imported and route attributes of the imported routes are modified to meet the requirements of the protocol.
- Modifies route attributes.
Attributes of the routes that match the specified route-filter can be modified as required.
- Flexibly updates the policies for advertising and accepting routes
XPL supports the paragraph-by-paragraph editing mode. Configurations can be performed in the paragraph editing UI and are committed together after the configurations of a paragraph are complete. Therefore, policy modification and update are relatively flexible.

Benefits

XPL offers the following benefits:

- Saves system resources by controlling the routing table size.
- Improves network security by controlling route advertisement and acceptance.
- Improves network performance by modifying route attributes for effective traffic planning.
- Simplifies routing policy configurations.

1.1.12.2 Configuration Precautions for XPL

Feature Requirements

Table 1-189 Feature requirements

Feature Requirements	Series	Models
If you enter the XPL line editing mode again, the new content overwrites the previous content. In line editing mode, you need to edit the entire policy content at a time.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/NetEngine 8000 X16/NetEngine 8000E X8/NetEngine 8100 X

Feature Requirements	Series	Models
To modify an existing global variable set, route attribute set, or route-filter using a command line, enter the specific command view to reconfigure the set or policy. Otherwise, the newly configured matching and application rules overwrite the historical matching and application rules. If historical matching and application rules need to be retained, you need to reconfigure and add historical matching and application rules.	NetEngine 8000 X	NetEngine 8000 X4/NetEngine 8000 X8/ NetEngine 8000 X16/NetEngine 8000E X8/ NetEngine 8100 X

1.1.12.2.3 Introduction to XPL Paragraph-by-Paragraph Editing

The paragraph-by-paragraph editing mode is a new XPL configuration mode. For the comparison between the paragraph-by-paragraph and line-by-line editing modes, see [1.1.12.2.1 Overview of XPL](#). The following uses [Common Shortcut Keys](#), [Example of Paragraph-by-Paragraph Editing Operations](#), and [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#) as examples to describe the paragraph-by-paragraph editing mode.

The paragraph-by-paragraph editing interface functions as a text editor, in which users can edit XPL clauses to configure or modify sets and route-filters. After the configuration is complete, it can be committed by stage.

Common Shortcut Keys

Shortcut keys can be used in the paragraph-by-paragraph editing interface. [Table 1-190](#) lists common shortcut keys and their functions.

Table 1-190 Shortcut keys commonly used in the paragraph-by-paragraph editing interface

Shortcut Key	Function
i	Enters the text editing mode. Sets or route-filters can be configured only in the text editing mode. The shortcut key is the i key on the keyboard.
Esc	Exits the text editing mode. The shortcut key is the Esc key in the upper left corner of the keyboard.
:q!	Exits the paragraph-by-paragraph editing view without saving configurations. You need to press Esc to exit the text editing mode, and then enter the shortcut key manually. NOTE The shortcut key takes effect only after Enter is pressed.

Shortcut Key	Function
:wq	<p>Saves the configuration and exits from the paragraph-by-paragraph editing view. You need to press Esc to exit the text editing mode, and then enter the shortcut key manually.</p> <p>NOTE The shortcut key takes effect only after Enter is pressed.</p>

Entering the paragraph-by-paragraph editing view

The following lists the different XPL paragraph-by-paragraph editing views and the commands used to access them:

- Global variable set: Run the **edit xpl global-value** command.
- IPv4 prefix set: Run the **edit xpl ip-prefix-list *ip-prefix-list-name*** command.
- IPv6 prefix set: Run the **edit xpl ipv6-prefix-list *ipv6-prefix-list-name*** command.
- AS_Path set: Run the **edit xpl as-path-list *as-path-list-name*** command.
- Community set: Run the **edit xpl community-list *community-list-name*** command.
- Large-Community set: Run the **edit xpl large-community-list *large-community-list-name*** command.
- Route-filter: Run the **edit xpl route-filter *route-filter-name*** command.

Example of Paragraph-by-Paragraph Editing Operations

Table 1-191 Description of paragraph-by-paragraph editing operations

No.	Item	Description	Picture
1	Enter the paragraph-by-paragraph editing view and the text editing mode.	<p>For example, run the edit xpl ip-prefix-list ip-prefix-list-name command in the user view to enter the IPv4 prefix set paragraph-by-paragraph editing view, and press i to enter the text editing mode. -- INSERT -- is displayed at the bottom of the interface, indicating that you can enter content, as shown in the figure on the right.</p> <p>If you do not press i, content cannot be entered.</p>	

No.	Item	Description	Picture
2	Configure XPL sets and route-filters.	<p>In text editing mode, configure a start clause xpl ip-prefix-list ip-prefix-list-name, an element, and an end clause end-list for the IPv4 prefix set, and press Esc to exit the text editing mode.</p> <p>After you press Esc, -- INSERT -- at the bottom of the interface disappears. In this case, you cannot enter any content. If you need to continue your editing, press i again to return to the text editing mode.</p> <p>Note that set elements must be separated by commas (,). Otherwise, an error is reported.</p>	<pre><deviceA>edit xpl ip-prefix-list prefixa xpl ip-prefix-list prefixa 2.2.2.0 24, 3.3.3.0 24, 4.4.4.0 24 end-list ~</pre> <p>NOTE</p> <ul style="list-style-type: none"> • In the preceding figure, xpl ip-prefix-list prefixa is the start clause, and end-list is the end clause. Both of them are mandatory. • After you run the edit xpl ip-prefix-list prefixa command to enter the paragraph-by-paragraph editing view, if edit xpl ip-prefix-list prefixb is entered as the start clause, the set elements of prefixb rather than prefixa are modified. Ensure that the input is correct.

No.	Item	Description	Picture
3	Save configurations and exit from the paragraph-by-paragraph editing view.	<p>After the configuration is complete, press Esc. -- INSERT -- at the bottom of the interface disappears, indicating that you have exited the text editing mode. In this case, you can enter :wq and press Enter to save the configuration and exit the paragraph-by-paragraph editing view. You can also enter :q! and press Enter to exit the paragraph-by-paragraph editing view without saving the configuration.</p> <ul style="list-style-type: none"> Save configurations and exit from the paragraph-by-paragraph editing view. A message is displayed, asking you whether to submit the configuration after the following operations: complete configuration, press Esc to exit the text editing mode, enter :wq, and press Enter. In this case, you can enter y to commit the configuration and press Enter to return to the user view. This indicates that the XPL configuration is saved successfully, as shown in the following figures. <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:y <DeviceA></pre> <p>If you enter n, the configuration is not saved. After you press Enter, the system returns to the user view too.</p> <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:n <DeviceA></pre> <p>If you enter c, the configuration is not submitted. After you press Enter, the system returns to the paragraph-by-paragraph editing view. In this case, you can press i again to enter the text editing mode.</p> <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:c <DeviceA></pre> <p>After you run the edit xpl ip-prefix-list ip-prefix-list-name command to enter the paragraph editing view, if you press i to enter the text editing mode and then press Esc to exit the text editing mode without entering any information, enter :wq, and press Enter, the system</p>	<ul style="list-style-type: none"> Save configurations and exit from the paragraph-by-paragraph editing view. A message is displayed, asking you whether to submit the configuration after the following operations: complete configuration, press Esc to exit the text editing mode, enter :wq, and press Enter. In this case, you can enter y to commit the configuration and press Enter to return to the user view. This indicates that the XPL configuration is saved successfully, as shown in the following figures. <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:y <DeviceA></pre> <p>If you enter n, the configuration is not saved. After you press Enter, the system returns to the user view too.</p> <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:n <DeviceA></pre> <p>If you enter c, the configuration is not submitted. After you press Enter, the system returns to the paragraph-by-paragraph editing view. In this case, you can press i again to enter the text editing mode.</p> <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:c <DeviceA></pre> <p>After you run the edit xpl ip-prefix-list ip-prefix-list-name command to enter the paragraph editing view, if you press i to enter the text editing mode and then press Esc to exit the text editing mode without entering any information, enter :wq, and press Enter, the system</p>

No.	Item	Description	Picture
			<p>displays a message asking you whether to continue editing the configuration instead of whether to commit the configuration. You need to enter n to return to the user view. Determine the subsequent operations based on the displayed information.</p> <ul style="list-style-type: none">• Exit the paragraph-by-paragraph editing view without saving the configuration. After the configuration is complete, press :q! and press Enter. A message is displayed, asking you whether to continue the editing. In this case, you can enter y to return to the paragraph-by-paragraph editing view, and then press i to enter the text editing mode. <p>Continue editing? [Y/N]:y</p> <pre>xpl ip-prefix-list prefixa 2.2.2.0 24, 3.3.3.0 24, 4.4.4.0 24 end-list</pre> <p>If you enter n, the system exits the paragraph-by-paragraph editing mode and returns to the user view. In this case, the configuration does not take effect.</p> <p>Continue editing? [Y/N]:n</p> <pre><DeviceA></pre>

No.	Item	Description	Picture
4	The configuration is incorrect and cannot be saved.	<p>If the configuration is incorrect, it cannot be submitted or saved. In this case, you can modify the configuration based on the error information and submit it again.</p> <p>After the configuration is complete, enter :wq to save the configuration and exit the paragraph-by-paragraph editing view. Press Enter. A message is displayed asking you whether to commit the configuration. Enter y to commit the configuration. If the configuration is incorrect, an error message is displayed, as shown in the figure on the right.</p> <p>In this case, you can continue to enter y to return to the paragraph-by-paragraph editing view. Press i again to enter the text editing mode. Based on the error information, use arrow keys on the keyboard to move the cursor to the corresponding position and modify the information.</p>	<ul style="list-style-type: none"> Incorrect configuration example 1 The configuration error is that the comma (,) is missing after 3.3.3.0 24. In this case, you can enter y to return to the paragraph-by-paragraph editing view, and then press i to enter the text editing mode. Press arrow keys to move the cursor to the end of 3.3.3.0 24, supplement a comma (,), and then submit the configuration. <pre><DeviceA>edit xpl ip-prefix-list prefixa xpl ip-prefix-list prefixa 2.2.2.0 24, 3.3.3.0 24 4.4.4.0 24 end-list■</pre> <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:y Syntax error in the text (Line:4): 4.4.4.0 24 Error: Unrecognized command found at '^' position. Continue editing? [Y/N]:</pre> <ul style="list-style-type: none"> Incorrect configuration example 2 The configuration error is invalid input. To exit the text editing mode, press Esc instead of typing esc. If you encounter this issue, enter y to return to the paragraph-by-paragraph editing view, and then press i to enter the text editing mode. Press arrow keys to move the cursor to esc and delete it, and then continue to submit the configuration. <pre>xpl ip-prefix-list prefixa 2.2.2.0 24, 3.3.3.0 24 4.4.4.0 24 end-list esc</pre> <pre>Proceed with commit (yes/no/cancel)? [Y/N/C]:y Syntax error in the text (Line:7): esc ^ Error: Unrecognized command found at '^' position. Continue editing? [Y/N]:■</pre>

No.	Item	Description	Picture
5	Change configuration.	<p>To modify the existing configuration, run the edit xpl ip-prefix-list ip-prefix-list-name command to enter the IPv4 prefix set paragraph-by-paragraph editing view and press i to enter the text editing mode.</p> <p>In this case, you can use arrow keys to move the cursor to the required position and delete or add set elements. Note that elements need to be separated by commas (,).</p> <p>After the modification, press Esc to exit the text editing mode. Enter :wq and press Enter to save the configuration and exit the paragraph-by-paragraph editing view.</p>	<ul style="list-style-type: none"> Delete the set element 3.3.3.0 24 and save the configuration. <pre>xpl ip-prefix-list prefixa 2.2.2.0 24, 4.4.4.0 24 end-list : : : :wq</pre> Add 5.5.5.0 24 to the original configuration and save the configuration. <pre>xpl ip-prefix-list prefixa 2.2.2.0 24, 3.3.3.0 24, 4.4.4.0 24, 5.5.5.0 24 end-list : : : :wq</pre>

Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode

Sets and route-filters can be configured as follows:

- To configure a global variable set:
 - Configure a start clause (**xpl global-value**) for a global variable set.
 - Configure set elements in the format of variable name+'value' in the global variable set view.
 - The variable value range is The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).The

variable name must not be **abort**, **display**, **end-global-value**, or any abbreviation of the keywords, for example, **a**, **ab**, **abo**, **di**, **e**, or **end**.

- Separate every two neighboring elements with a comma (,), for example, aaa '12', bbb '34', aaa '1.2.3.4'.
- c. Configure an end clause (**end-global-value**) for the global variable set.
- To configure an IPv4 prefix set:
 - a. Configure a start clause (**xpl ip-prefix-list ip-prefix-list-name**) for an IPv4 prefix set.
 - b. Configure elements (IPv4 addresses with masks, 1.1.1.0 24 for example) for the set and separate every two neighboring elements with a comma (,), You can use **eq**, **ge**, or **le** to specify the mask length. For example, **1.1.1.0 24 ge 26 le 30** matches the routes in network segment 1.1.1.0/24 with the mask length ranging from 26 to 30 bits.
 - c. Configure an end clause (**end-list**) for the IPv4 prefix set.
- To configure an IPv6 prefix set:
 - a. Configure a start clause (**xpl ipv6-prefix-list ipv6-prefix-list-name**) for an IPv6 prefix set.
 - b. Configure elements (IPv6 addresses with masks, 2001:db8:0:1:: 64 for example) for the set and separate every two neighboring elements with a comma (,), You can use **eq**, **ge**, or **le** to specify the mask length. For example, **2001:db8:0:1:: 64 ge 96 le 100** matches the routes in network segment 2001:db8:0:1::/64 with the mask length ranging from 96 to 100 bits.
 - c. Configure an end clause (**end-list**) for the IPv6 prefix set.
- To configure an AS_Path set:
 - a. Configure a start clause (**xpl as-path-list as-path-list-name**) for an AS_Path set.
 - b. Configure elements for the set and separate every two neighboring elements with a comma (,), The elements can be configured in any of the following formats:
 - **length { eq | ge | le } as-length**: matches BGP routes with AS_Path length equal to (**eq**), greater than or equal to (**ge**), or less than or equal to (**le**) *as-length*. The value of *as-length* is an integer ranging from 0 to 2047.
 - **unique-length { eq | ge | le } as-length**: matches BGP routes with AS_Path length equal to (**eq**), greater than or equal to (**ge**), or less than or equal to (**le**) *as-length* (duplicate AS numbers are counted as one). The value of *as-length* is an integer ranging from 0 to 2047.
 - **origin as-path [whole-match]**: matches BGP routes with AS_Path whose rightmost AS numbers are the same as *as-path*. The *as-path* parameter is enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. Duplicate AS numbers are counted as one unless **whole-match** is configured.
 - **peer-is as-path [whole-match]**: matches BGP routes with AS_Path whose leftmost AS numbers are the same as *as-path*. The *as-path* parameter is enclosed in single quotation marks, with every two

neighboring AS numbers separated with a space. Duplicate AS numbers are counted as one unless **whole-match** is configured.

- **pass as-path [whole-match]:** matches BGP routes with AS_Path whose contiguous AS numbers match *as-path*. The *as-path* parameter is enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. Duplicate AS numbers are counted as one unless **whole-match** is configured.
- **regular regular-expression:** matches BGP routes with AS_Path in the specified regular expression.

 **NOTE**

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- c. Configure an end clause (**end-list**) for the AS_Path set.
- To configure a community set:
 - a. Configure a start clause (**xpl community-list *community-list-name***) for a community set.
 - b. Configure elements in the format of aa:nn (100:1 for example), a community number, or a known community (**internet**, **no-export-subconfed**, **no-advertise**, or **no-export**) for the community set and separate every two neighboring elements with a comma (,). Alternatively, configure elements in the format of **regular regular-expression**, which matches routes with community attributes in the specified regular expression.

 **NOTE**

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

The community-based *regular expression* can be set to a character string in either the aa:nn format or integer format. The following are two examples:

The regular ^1:1\$ configuration matches routes that carry the community value of 65537 or 1:1.

The regular ^65537\$ configuration also matches routes that carry the community value of 65537 or 1:1.

- c. Configure an end clause (**end-list**) for the community set.
- Large-Community set:
 - a. Configure a start clause (**xpl large-community-list *large-community-list-name***) for the Large-Community set.

- b. Configure elements in the format of aa:bb:cc (100:1:1 for example) for the Large-Community set and separate every two neighboring elements with a comma (,). Alternatively, configure elements in the format of **regular regular-expression**, which matches routes with large-community attribute in the specified regular expression.

 NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- c. Configure an end clause (**end-list**) for the Large-Community set.
- To configure a route-filter:
 - a. Configure a start clause in the format of **xpl route-filter route-filter-name(\$var1,\$var2,...)** for a route-filter. A maximum of eight parameters can be configured in a start clause, and the parameters can be used in condition or action clauses.
 - b. Configure a condition clause in the format of **if+condition clause+then** and connect the conditions in the clause with the Boolean operator **NOT**, **AND**, or **OR**.

 NOTE

Route-filters can have only action clauses and can also be empty (configured with only a start clause and an end clause). In this case, the default action **refuse** is used. If an empty route-filter is specified in another route-filter using a **call** clause, the empty route-filter does not take effect.

- c. Configure an action clause.

 NOTE

- Multiple action clauses can be configured if they do not conflict with each other.
- Action clauses (excluding **approve**, **refuse**, **finish**, **break**, and **call route-filter route-filter-name**) must follow **apply**.

- d. (Optional) Configure **elseif+condition clause+then** to filter the routes that fail to match the conditions specified in the **if** clause and specify an action clause for the **elseif** clause. You can configure multiple **elseif** clauses to filter the routes that fail to meet the previous matching rule or configure an **else** clause to match the routes that fail to meet all the previous matching rules. Each **if**, **elseif**, or **else** clause must be followed by an action clause.
- e. Configure a conclusive condition clause (**endif**).

NOTE

Steps 2 to 5 describe how to configure an **if** condition branch. One route-filter can have multiple **if** condition branches, and the **if** condition branches can be configured as follows:

- One **if** condition branch is followed by another.
- The **if+condition clause+then** or **elseif+condition clause+then** is followed by another **if** condition branch. Such a configuration further filters routes that match **if+condition clause+then** or **elseif+condition clause+then** against the second **if** condition branch.

Regardless of the configuration mode, route filtering continues until **finish**, **break**, **refuse**, or the last **if** condition branch.

- f. Configure an end clause (**end-filter**) for the route-filter.

NOTE

An easy method to configure route-filters to reference route attribute sets is to use the format {element A, element B...}, **if ip route-source in { 1.1.1.0 24, 2.2.2.2 32 }** **then** for example. However, if a route-filter needs to reference a set multiple times, configure named route attribute sets. You can choose a configuration method as required.

Purpose

When advertising, receiving, or importing routes, the router can use XPL based on actual networking requirements to filter routes and modify route attributes. XPL serves the following purposes:

- Controls route advertisement.
Only matched routes are advertised.
- Controls route acceptance.
Only necessary and valid routes are accepted, which reduces the routing table size and improves network security.
- Filters and controls imported routes.
A routing protocol may import routes discovered by other routing protocols. XPL ensures that only the routes that meet certain conditions are imported and route attributes of the imported routes are modified to meet the requirements of the protocol.
- Modifies route attributes.
Attributes of the routes that match the specified route-filter can be modified as required.
- Flexibly updates the policies for advertising and accepting routes
XPL supports the paragraph-by-paragraph editing mode. Configurations can be performed in the paragraph editing UI and are committed together after the configurations of a paragraph are complete. Therefore, policy modification and update are relatively flexible.

Benefits

XPL offers the following benefits:

- Saves system resources by controlling the routing table size.

- Improves network security by controlling route advertisement and acceptance.
- Improves network performance by modifying route attributes for effective traffic planning.
- Simplifies routing policy configurations.

1.1.12.2.4 Configuring a Global Variable Set

A global variable set is a group of frequently used values that are defined as global variables. Global variables can be referenced by all route-filters configured on a device.

Context

NOTE

The data type of each value in a global variable set may vary with the route-filter that references the set.

The global variables on a device must be unique. A new global variable will override an existing global variable with the same name.

To enable a route-filter or a set to reference a global variable, enter \$+variable name, for example, **\$globvar**.

Procedure

- Configure a global variable set using the paragraph-by-paragraph editing mode.
 - a. Run the **edit xpl global-value** command to enter the global variable set paragraph-by-paragraph editing view.
 - b. Press **i** to enter the text editing mode.

 NOTE

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

 - c. Configure a start clause (**xpl global-value**) for a global variable set.
 - d. Configure set elements in the format of variable name+'value', for example, aaa '12', bbb'34', aaa '1.2.3.4'. Separate every two neighboring elements with a comma (,).
 - e. Configure an end clause (**end-global-value**) for the global variable set.
 - f. Press **Esc** to exit from the text editing mode.
 - g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure a global variable set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.

- b. Run the **xpl global-value** command to enter the global variable set view.
- c. Configure elements in the format of variable name+'value' in the global variable set view, for example, aaa'12', bbb'34', and aaa'1.2.3.4'. Separate every two neighboring elements with a comma (,).
- d. Run the **end-global-value** command to conclude the configuration of the global variable set.
- e. Run the **commit** command to commit the configuration.

----End

Example



If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure a global variable set that includes two values: 100 and ip-prefix.
Configuration Example	<pre><HUAWEI> edit xpl global-value xpl global-value aaa '100', bbb 'ip-prefix', end-global-value</pre> <p>The global variable set includes a set named aaa with value 100 and another set named bbb with value ip-prefix.</p>
Objective	Configure a route-filter to permit only the routes with the MED value less than 100 and the routes that match the IPv4 prefix set named bbb .

Reference Example	<pre><HUAWEI> edit xpl ip-prefix-list r1 xpl route-filter r1 if ip med le \$aaa then approve elseif ip route-destination in \$bbb then approve else refuse endif end-filter</pre> <p>The route-filter uses two global variables (aaa and bbb), in which aaa represents a MED value and bbb represents an IPv4 destination address prefix set. The route-filter permits only the routes with the MED value less than 100 and the routes that match bbb.</p>
--------------------------	---

1.1.12.2.5 Configuring a Route Attribute Set

Route attribute sets can be referenced by route-filters to filter the routes with the same or similar route attribute.

Usage Scenario

If the routes to be filtered have the same or similar route attribute, you can configure a route attribute set for them, and the route attribute set can be referenced by a route-filter to filter the routes. The route attribute sets supported by the NetEngine 8100 X, NetEngine 8000 X and NetEngine 8000E X are as follows:

- IPv4 prefix sets: apply to all dynamic routing protocols and can be used to match source, destination, and next hop IP addresses.
- IPv6 prefix sets: apply to all dynamic routing protocols and can be used to match source, destination, and next hop IPv6 addresses.
- AS_Path sets: apply only to BGP and are used to match the AS_Path attribute of BGP routes.
- Community sets: apply only to BGP and are used to match the community attribute of BGP routes.
- RD sets: match the RD attributes of VPN routes.
- Route target sets: match the RTs of VPN routes.
- Site of origin (SoO) sets: match the SoOs of VPN routes.

NOTE

Sets do not have the permit or deny function as routing policies do. Instead, sets are only groups of data used as matching rules, and the actions to be applied are specified in route-filters.

Route attribute sets can have no elements.

The easiest method to configure route-filters to reference route attribute sets is to use the format {element A, element B...}, **if ip route-source in { 1.1.1.0 24, 2.2.2.2 32 } then** for example. However, if a route-filter needs to reference a set multiple times, configure named route attribute sets.

Pre-configuration Tasks

Before configuring a route attribute set, complete the following tasks:

- Select a route attribute set based on the characteristics of the routes to be filtered.
- If some values need to be frequently used, configure a global variable set, which is easy to configure and use.

Perform one or more of the following configurations as required:

Configuring an IPv4 Prefix Set

IPv4 prefix sets apply to all dynamic routing protocols and can be used to match source, destination, and next hop IP addresses.

Procedure

- Configure an IPv4 prefix set using the paragraph-by-paragraph editing mode.
 - a. Run the **edit xpl ip-prefix-list ip-prefix-list-name** command to enter the IPv4 prefix set paragraph-by-paragraph editing view.
 - b. Press **i** to enter the text editing mode.

NOTE

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- c. Configure a start clause (**xpl ip-prefix-list ip-prefix-list-name**) for an IPv4 prefix set.
- d. Configure elements (IPv4 addresses with masks, 1.1.1.0 24 for example) for the set and separate every two neighboring elements with a comma (,). You can use **eq**, **ge**, or **le** to specify the mask length.
- e. Configure an end clause (**end-list**) for the IPv4 prefix set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure an IPv4 prefix set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl ip-prefix-list ip-prefix-list-name** command to enter the IPv4 prefix set view.
 - c. Configure elements (IPv4 addresses with masks, such as 1.1.1.0 24) for the set and separate every two neighboring elements with a comma (,). You can use **eq**, **ge**, or **le** to specify the mask length.
 - d. Run the **end-list** command to conclude the configuration of the IPv4 prefix set, exit the IPv4 prefix set view, and return to the system view.
 - e. Run the **commit** command to commit the configuration.

----End

Example

NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure an IPv4 prefix set to match the routes in 1.1.1.0/24, 2.2.2.0/24, and 3.3.3.3/32 network segments.
Configuration Example 1	<pre><HUAWEI> edit xpl ip-prefix-list aaa xpl ip-prefix-list aaa 1.1.1.0 24, 2.2.2.0 24, 3.3.3.3 32 end-list</pre> <p>This prefix set includes three elements and matches the routes in 1.1.1.0/24, 2.2.2.0/24, and 3.3.3.3/32 network segments.</p>
Objective	Configure an IPv4 prefix set to match the routes in network segment 1.1.1.0/24 with the mask length ranging from 26 to 30 bits, the routes in network segment 2.2.2.0/24 with the mask length greater than or equal to 28 bits, and the routes in network segment 3.3.3.0/24 with the mask length of 30 bits.

Configuration Example 2	<pre><HUAWEI> edit xpl ip-prefix-list bbb xpl ip-prefix-list bbb 1.1.1.0 24 ge 26 le 30, 2.2.2.0 24 ge 28, 3.3.3.0 24 eq 30 end-list</pre> <p>This IPv4 prefix set includes three elements and matches the routes in network segment 1.1.1.0/24 with the mask length ranging from 26 to 30 bits, the routes in network segment 2.2.2.0/24 with the mask length greater than or equal to 28 bits, and the routes in network segment 3.3.3.0/24 with the mask length of 30 bits.</p>
Objective	Configure a route-filter to set MED 60 for the routes that match IPv4 prefix set bbb , set MED 70 for the routes that do not match IPv4 prefix set bbb but match aaa , and set MED 80 for all other routes.
Reference Example	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if ip route-destination in bbb then apply med 60 elseif ip route-destination in aaa then apply med 70 else apply med 80 endif end-filter</pre> <p>The route-filter references two IPv4 destination address prefix sets (aaa and bbb) and sets MED 60 for the routes that match the set bbb, sets MED 70 for the routes that do not match the set bbb but match the set aaa, and sets MED 80 for all other routes.</p>

Configuring an IPv6 Prefix Set

IPv6 prefix sets apply to all dynamic routing protocols and can be used to match source, destination, and next hop IPv6 addresses.

Procedure

- Configure an IPv6 prefix set using the paragraph editing mode.
 - a. Run the **edit xpl ipv6-prefix-list *ipv6-prefix-list-name*** command to enter the IPv6 prefix set paragraph editing interface view.
 - b. Press **i** to enter the text editing mode.

 NOTE

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- c. Configure a start clause (**xpl ipv6-prefix-list** *ipv6-prefix-list-name*) for an IPv6 prefix set.
- d. Configure elements (IPv6 addresses with masks, 2001:db8:0:1::64 for example) for the set and separate every two neighboring elements with a comma (,). You can use **eq**, **ge**, or **le** to specify the mask length.
- e. Configure an end clause (**end-list**) for the IPv6 prefix set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure an IPv6 prefix set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl ipv6-prefix-list** *ipv6-prefix-list-name* command to enter the IPv6 prefix set view.
 - c. Configure elements (IPv6 addresses with masks, such as 2001:db8:0:1::64) for the set and separate every two neighboring elements with a comma (,). You can use **eq**, **ge**, or **le** to specify the mask length.
 - d. Run the **end-list** command to conclude the configuration of the IPv6 prefix set, exit the IPv6 prefix set view, and return to the system view.
 - e. Run the **commit** command to commit the configuration.

----End

Example

 NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure an IPv6 prefix set to match the routes in network segments 2001:db8:0:1::/64, 2001:db8:1:1::/64, and 2001:db8:1:2::/64.
------------------	---

Configuration Example 1	<pre><HUAWEI> edit xpl ipv6-prefix-list aaa xpl ipv6-prefix-list aaa 2001:db8:0:1:: 64, 2001:db8:1:1:: 64, 2001:db8:1:2:: 64 end-list</pre> <p>This IPv6 prefix set includes three elements and matches the routes in network segments 2001:db8:0:1::/64, 2001:db8:1:1::/64, and 2001:db8:1:2::/64.</p>
Objective	Configure an IPv6 prefix set to match the routes in network segment 2001:db8:0:1::/64 with the mask length ranging from 96 to 100 bits, the routes in network segment 2001:db8:1:1::/64 with the mask length less than 100 bits, and the routes in network segment 2001:db8:1:2::/64 with the mask length of 96 bits.
Configuration Example 2	<pre><HUAWEI> edit xpl ipv6-prefix-list bbb xpl ipv6-prefix-list bbb 2001:db8:0:1:: 64 ge 96 le 100, 2001:db8:1:1:: 64 le 100, 2001:db8:1:2:: 64 eq 96 end-list</pre> <p>This IPv6 prefix set includes three elements and matches the routes in network segment 2001:db8:0:1::/64 with the mask length ranging from 96 to 100 bits, the routes in network segment 2001:db8:1:1::/64 with the mask length less than 100 bits, and the routes in network segment 2001:db8:1:2::/64 with the mask length of 96 bits.</p>
Objective	Configure a route-filter to deny the routes that match set bbb and the routes that match neither set aaa nor bbb and permit the routes that do not match bbb but match aaa and increase their MED values by 10.

Reference Example	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if ip route-destination in bbb then refuse elseif ip route-destination in aaa then apply med + 10 else refuse endif end-filter</pre> <p>The route-filter references two IPv6 destination address prefix sets (aaa and bbb), denies the routes that match IPv6 destination address prefix set bbb, permits the routes that do not match bbb but match aaa and increases their MED values by 10, and denies all other routes.</p>
--------------------------	---

Configuring an AS_Path Set

AS_Path sets apply only to BGP and are used to match the AS_Path attribute of BGP routes. This section describes how to configure AS_Path sets.

Procedure

- Configure an AS_Path set using the paragraph editing mode.
 - Run the **edit xpl as-path-list as-path-list-name** command to enter the AS_Path set paragraph editing interface view.
 - Press **i** to enter the text editing mode.

 **NOTE**

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- Configure a start clause (**xpl as-path-list as-path-list-name**) for an AS_Path set.
- Configure elements in the format of regular+AS_Path regular expression for the set and separate every two neighboring elements with a comma (,). For example, regular 1_2_3_4 represents an AS_Path that contains AS numbers 1, 2, 3, and 4.

 **NOTE**

Regular expression processing is CPU computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- Configure an end clause (**end-list**) for the AS_Path set.

- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure an AS_Path set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl as-path-list *as-path-list-name*** command to enter the AS_Path set view.
 - c. Configure elements in the format of regular+AS_Path regular expression for the set and separate every two neighboring elements with a comma (,). For example, regular 1_2_3_4 represents an AS_Path that contains AS numbers 1, 2, 3, and 4.

 NOTE

Regular expression processing is CPU computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

- d. Run the **end-list** command to conclude the configuration of the AS_Path set, exit the AS_Path set view, and return to the system view.
- e. Run the **commit** command to commit the configuration.

----End

Example

 NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure an AS_Path set to match the routes received from AS 100, the routes that pass through AS 200, and the routes that originate from AS 300.
------------------	--

Configuration Example	<pre><HUAWEI> edit xpl as-path-list aaa xpl as-path-list aaa regular ^100_ regular _200_ regular _300\$ end-list</pre> <p>This AS_Path set includes three elements and matches the routes received from AS 100, the routes that pass through AS 200, and the routes that originate from AS 300.</p>
Objective	Configure a route-filter to deny the routes received from AS 100, the routes that pass through AS 200, and the routes that originate from AS 300.
Reference Example	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if as-path in aaa then refuse else approve endif end-filter</pre> <p>The route-filter references AS_Path set aaa, denies the routes that match aaa, and permits all other routes.</p>

Configuring a Community Set

Community sets apply only to BGP and are used to match the community attribute of BGP routes. This section describes how to configure community sets.

Procedure

- Configure a community set using the paragraph editing mode.
 - a. Run the **edit xpl community-list *community-list-name*** command to enter the community set paragraph editing interface view.
 - b. Press **i** to enter the text editing mode.

NOTE

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- c. Configure a start clause (**xpl community-list *community-list-name***) for a community set.
- d. Configure elements in the format of aa:nn (100:1 for example), a community number, or a known community attribute (**internet**, **no-export-subconfed**, **no-advertise**, or **no-export**) for the community set and separate every two neighboring elements with a comma (,). You can also configure elements in the **regular regular-expression** format to indicate that the community attribute of a route matches a specified

regular expression, and *regular-expression* specifies a regular expression. An asterisk (*) in an element can be used to match any digit. For example, 100:* indicates that the element matches all community values with the AS number 100.

 NOTE

Regular expression processing is CPU computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- e. Configure an end clause (**end-list**) for the community set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure a community set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl community-list *community-list-name*** command to enter the community set view.
 - c. Configure elements in the format of aa:nn (100:1 for example), a community number, or a known community attribute (**internet**, **no-export-subconfed**, **no-advertise**, or **no-export**) for the community set and separate every two neighboring elements with a comma (,). You can also configure elements in the **regular regular-expression** format to indicate that the community attribute of a route matches a specified regular expression, and *regular-expression* specifies a regular expression. An asterisk (*) in an element can be used to match any digit. For example, 100:* indicates that the element matches all community values with the AS number 100.

 NOTE

Regular expression processing is CPU computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- d. Run the **end-list** command to conclude the configuration of the community set, exit the community set view, and return to the system view.

- e. Run the **commit** command to commit the configuration.

----End

Example

NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure a community set to match the routes carrying community set 100:1, 200:1, 300:1, or no-export.
Configuration Example 1	<pre><HUAWEI> edit xpl community-list aaa xpl community-list aaa 100:1, 200:1, 300:1, no-export end-list</pre> <p>This community set includes four elements and matches the routes carrying community set 100:1, 200:1, 300:1, or no-export.</p>
Objective	Configure a community set to match the routes carrying community set 100:*, 200:*, or 300:1.
Configuration Example 2	<pre><HUAWEI> edit xpl community-list bbb xpl community-list bbb regular 100:*,* regular 200:*,* 300:1 end-list</pre> <p>This community set includes three elements and matches the routes carrying community set 100:*, 200:*, or 300:1.</p>
Objective	Configure a route-filter to add communities 100:1, 200:1, 300:1, and no-export to the routes with the Local_Pref value less than 100.

Reference Example 1	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if local-preference le 100 then apply community aaa additive endif end-filter</pre> <p>This route-filter references community set aaa and adds communities 100:1, 200:1, 300:1, and no-export to the routes with the Local_Pref value less than or equal to 100.</p>
Objective	Configure a route-filter to set the next hop addresses of the routes whose communities are a subset of the community set bbb to 1.1.1.1.
Reference Example 2	<pre><HUAWEI> edit xpl route-filter r2 xpl route-filter r2 if community matches-within bbb then apply ip next-hop 1.1.1.1 endif end-filter</pre> <p>This route-filter references community set bbb and sets the next hop addresses of the routes whose communities are a subset of bbb to 1.1.1.1.</p>

Configuring Large-Community Sets

Large-Community sets apply to BGP and are used to match the Large-Community attribute of BGP routes. This section describes how to configure Large-Community sets.

Procedure

- Configure a Large-Community set using the paragraph-by-paragraph editing mode.
 - a. Run the **edit xpl large-community-list *large-community-list-name*** command to enter the paragraph editing interface view of a Large-Community set.
 - b. Press **i** to enter the text editing mode.



Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- c. Configure a start clause (**xpl large-community-list *large-community-list-name***) for the Large-Community set.
- d. Configure elements in the format of aa:bb:cc (100:1:1 for example) for the Large-Community set and separate every two neighboring elements

with a comma (,). You can also configure elements in the **regular regular-expression** format to indicate that the Large-Community attribute of a route matches a specified regular expression, and *regular-expression* specifies a regular expression. An asterisk (*) in an element can be used to match any digit; for example, an element can be set to 100:1:*.

 NOTE

Regular expression processing is CPU computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- e. Configure an end clause (**end-list**) for the Large-Community set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure a Large-Community set using the line-by-line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl large-community-list *large-community-list-name*** command to enter the Large-Community set view.
 - c. Configure elements in the format of aa:bb:cc (100:1:1 for example) for the Large-Community set and separate every two neighboring elements with a comma (,). You can also configure elements in the **regular regular-expression** format to indicate that the Large-Community attribute of a route matches a specified regular expression, and *regular-expression* specifies a regular expression. An asterisk (*) in an element can be used to match any digit; for example, an element can be set to 100:1:*.

 NOTE

Regular expression processing is CPU computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- d. Run the **end-list** command to conclude the configuration of the Large-Community set, exit the Large-Community set view, and return to the system view.

- e. Run the **commit** command to commit the configuration.

----End

Example

NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure a Large-Community set to match Large-Community attribute values 100:1:1, 200:1:1, and 300:1:1.
Configuration Example 1	<pre><HUAWEI> edit xpl large-community-list aaa xpl large-community-list aaa 100:1:1, 200:1:1, 300:1:1 end-list</pre> <p>This Large-Community set includes three elements and matches Large-Community attribute values 100:1:1, 200:1:1, and 300:1:1.</p>
Objectives	Configure a Large-Community set to match Large-Community attribute values 100:1:*, 200:1:*, and 300:1:1.
Configuration Example 2	<pre><HUAWEI> edit xpl large-community-list bbb xpl large-community-list bbb regular 100:1:* regular 200:1:* 300:1:1 end-list</pre> <p>This Large-Community set includes three elements and matches Large-Community attribute values 100:1:*, 200:1:*, and 300:1:1.</p>
Objectives	Configure a route-filter to add Large-Community attribute values 100:1:1, 200:1:1, and 300:1:1 to the routes with a Local_Pref value less than or equal to 100.

Reference Example 1	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if local-preference le 100 then apply large-community aaa additive endif end-filter</pre> <p>This route-filter references the Large-Community set aaa and adds Large-Community attribute values 100:1:1, 200:1:1, and 300:1:1 to the routes with a Local_Pref value less than or equal to 100.</p>
Objectives	Configure a route-filter to set the next hop addresses of the routes whose Large-Community attribute values compose a subset of the set bbb to 1.1.1.1.
Reference Example 2	<pre><HUAWEI> edit xpl route-filter r2 xpl route-filter r2 if large-community matches-within bbb then apply ip next-hop 1.1.1.1 endif end-filter</pre> <p>This route-filter references the Large-Community set bbb and sets the next hop addresses of the routes whose Large-Community attribute values compose a subset of the set bbb to 1.1.1.1.</p>

Configuring an RD Set

RD sets are used to match the RD attributes of VPN routes.

Procedure

- Configure an RD set using the paragraph editing mode.
 - a. Run the **edit xpl rd-list rd-list-name** command to enter the RD set paragraph editing interface view.
 - b. Press **i** to enter the text editing mode.

 NOTE

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.
 - c. Configure a start clause (**xpl rd-list rd-list-name**) for an RD set.
 - d. Configure elements for the RD set and separate every two neighboring elements with a comma (,). The elements can be configured in any of the following formats:

- 16-bit AS number:32-bit user-defined number. For example, 101:3. The AS number ranges from 0 to 65535, and the user-defined number ranges from 0 to 4294967295.
- Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. An AS number ranges from 0 to 4294967295. A user-defined number ranges from 0 to 65535.
- 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. A 4-byte AS number in dotted notation is in the format of x.y, where x and y are integers that range from 1 to 65535 and from 0 to 65535, respectively. A user-defined number ranges from 0 to 65535.
- 32-bit IP address:16-bit user-defined number. For example, 192.168.122.15:1. The IP address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number ranges from 0 to 65535.
- **regular regular-expression**: matches VPN routes with RDs in the specified regular expression.

 NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- e. Configure an end clause (**end-list**) for the RD set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure an RD set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl rd-list rd-list-name** command to enter the RD set view.
 - c. Configure elements for the RD set and separate every two neighboring elements with a comma (,). The elements can be configured in any of the following formats:
 - 16-bit AS number:32-bit user-defined number. For example, 101:3. The AS number ranges from 0 to 65535, and the user-defined number ranges from 0 to 4294967295.

- Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. An AS number ranges from 0 to 4294967295. A user-defined number ranges from 0 to 65535.
- 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. A 4-byte AS number in dotted notation is in the format of x.y, where x and y are integers that range from 1 to 65535 and from 0 to 65535, respectively. A user-defined number ranges from 0 to 65535.
- 32-bit IP address:16-bit user-defined number. For example, 192.168.122.15:1. The IP address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number ranges from 0 to 65535.
- **regular regular-expression**: matches VPN routes with RDs in the specified regular expression.

 NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- d. Run the **end-list** command to conclude the configuration of the RD set, exit the RD set view, and return to the system view.
- e. Run the **commit** command to commit the configuration.

----End

Example

 NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For detailed description about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure an RD set to match the VPN routes carrying the RD (100:1, 200:1, or 300:*) in the format of 2-byte AS number:4-byte user-defined number.
------------------	--

Configuration Example	<pre><HUAWEI> edit xpl rd-list rd-list1 xpl rd-list rd-list1 100:1, 200:1, regular 300:* end-list</pre> <p>The RD set contains three elements and can match VPN routes carrying RD 100:1, 200:1, or 300:*. </p>
Objective	Configure a route-filter to allow only the VPN routes carrying the RDs contained in the set named rd-list1 to match the route-filter.
Reference Example	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if rd in rd-list1 then approve else refuse endif end-filter</pre> <p>The route-filter references the set named rd-list1 and allows only the VPN routes carrying the RDs contained in the set to match the route-filter.</p>

Configuring a Route Target Set

Route target sets are used to match the RTs of VPN routes.

Procedure

- Configure a route target set using the paragraph editing mode.
 - a. Run the **edit xpl extcommunity-list rt rt-list-name** command to enter the route target set paragraph editing interface view.
 - b. Press **i** to enter the text editing mode.
-  **NOTE**
- Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.
- c. Configure a start clause (**xpl extcommunity-list rt rt-list-name**) for the route target set.
 - d. Configure elements for the route target set and separate every two neighboring elements with a comma (,). The elements can be configured in any of the following formats:
 - 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. The AS

number and user-defined number must not be both 0s. Specifically, the RT must not be 0:0.

- IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535.
- Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is an integer ranging from 0 to 65535.
- 4-byte AS number in dotted notation ($x.y$):2-byte user-defined number, for example, 0.0:3 or 0.1:0. The x , y , and user-defined number are integers ranging from 0 to 65535. The AS number and user-defined number must not be both 0s. Specifically, the RT must not be 0:0.
- **regular regular-expression**: matches VPN routes with RTs in the specified regular expression.

NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- e. Configure an end clause (**end-list**) for the route target set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure a route target set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl extcommunity-list rt rt-list-name** command to enter the route target set view.
 - c. Configure elements for the route target set and separate every two neighboring elements with a comma (,). The elements can be configured in any of the following formats:
 - 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. The AS

number and user-defined number must not be both 0s. Specifically, the RT must not be 0:0.

- IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535.
- Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is an integer ranging from 0 to 65535.
- 4-byte AS number in dotted notation ($x.y$):2-byte user-defined number, for example, 0.0:3 or 0.1:0. The x , y , and user-defined number are integers ranging from 0 to 65535. The AS number and user-defined number must not be both 0s. Specifically, the RT must not be 0.0:0.
- **regular regular-expression:** matches VPN routes with RTs in the specified regular expression.

NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- d. Run the **end-list** command to conclude the configuration of the route target set, exit the route target set view, and return to the system view.
- e. Run the **commit** command to commit the configuration.

----End

Example

NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For detailed description about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure a route target set to match the VPN routes with the RT 100:1, 200:1, or 300:1.
------------------	--

Configuration Example	<pre><HUAWEI> edit xpl extcommunity-list rt rt-list1 xpl extcommunity-list rt rt-list1 100:1, 200:1, 300:1 end-list</pre> <p>The route target set contains three elements and can match VPN routes carrying RT 100:1, 200:1, or 300:1.</p>
Objective	Configure a route-filter to set the next hop IP addresses of the VPN routes carrying the RTs that are a subset of the set named rt-list1 and the VPN routes carrying the RTs of which the set named rt-list1 is a subset to 1.1.1.1 and 2.2.2.2, respectively.
Reference Example	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if extcommunity rt matches-within rt-list1 then apply ip next-hop 1.1.1.1 elseif extcommunity rt matches-all rt-list1 then apply ip next-hop 2.2.2.2 endif end-filter</pre> <p>The route-filter references the set named rt-list1 and sets the next hop IP addresses of the VPN routes carrying the RTs that are a subset of this set and the VPN routes carrying the RTs of which this set is a subset to 1.1.1.1 and 2.2.2.2, respectively.</p>

Configuring an SoO Set

Site of origin (SoO) sets are used to match the SoOs of VPN routes.

Procedure

- Configure an SoO set using the paragraph editing mode.
 - a. Run the **edit xpl extcommunity-list soo soo-list-name** command to enter the SoO set paragraph editing interface view.
 - b. Press **i** to enter the text editing mode.



Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- c. Configure a start clause (**xpl extcommunity-list soo soo-list-name**) for an SoO set.

- d. Configure elements for the SoO set and separate every two neighboring elements with a comma (,). The elements can be configured in any of the following formats:
 - 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. The AS number and user-defined number must not be both 0s. Specifically, the SoO must not be 0:0.
 - IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535.
 - Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is an integer ranging from 0 to 65535.
 - 4-byte AS number in dotted notation (*x.y*):2-byte user-defined number, for example, 0.0:3 or 0.1:0. The *x*, *y*, and user-defined number are integers ranging from 0 to 65535. The AS number and user-defined number must not be both 0s. Specifically, the SoO must not be 0.0:0.
 - **regular regular-expression**: matches VPN routes with SoO in the specified regular expression.

 NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- e. Configure an end clause (**end-list**) for the SoO set.
- f. Press **Esc** to exit from the text editing mode.
- g. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure an SoO set using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl extcommunity-list soo *soo-list-name*** command to enter the SoO set view.

- c. Configure elements for the SoO set and separate every two neighboring elements with a comma (,). The elements can be configured in any of the following formats:
 - 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. The AS number and user-defined number must not be both 0s. Specifically, the SoO must not be 0:0.
 - IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535.
 - Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is an integer ranging from 0 to 65535.
 - 4-byte AS number in dotted notation ($x.y$):2-byte user-defined number, for example, 0.0:3 or 0.1:0. The x , y , and user-defined number are integers ranging from 0 to 65535. The AS number and user-defined number must not be both 0s. Specifically, the SoO must not be 0.0:0.
 - **regular regular-expression**: matches VPN routes with SoO in the specified regular expression.

 NOTE

Regular expression processing is computing-intensive. When a large number of regular expressions are configured in an XPL policy to match a BGP route attribute and the length of the route attribute is long, the processing performance of the XPL policy deteriorates. To improve the processing performance of the routing policy, decrease the number of regular expressions or use a non-regular expression matching command.

It is recommended that a maximum of 100 regular expressions be configured for each policy.

- d. Run the **end-list** command to conclude the configuration of the SoO set, exit the SoO set view, and return to the system view.
- e. Run the **commit** command to commit the configuration.

----End

Example

 NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For detailed description about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objectives	Configure an SoO set to match the VPN routes carrying the SoO (100:1, 200:1, or 300:1).
Configuration Example	<pre><HUAWEI> edit xpl extcommunity-list soo soo-list1 xpl extcommunity-list soo soo-list1 100:1, 200:1, 300:1 end-list</pre> <p>The SoO set contains three elements and can match VPN routes carrying SoO 100:1, 200:1, or 300:1.</p>
Objective	Configure a route-filter to overwrite the SoOs of the VPN routes carrying at least one of the SoOs specified in the set named soo-list1 with 100:1 and 200:1.
Reference Example	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if extcommunity soo matches-any soo-list1 then apply extcommunity soo { 100:1,200:1 } overwrite endif end-filter</pre> <p>The route-filter references the set named soo-list1 and overwrites the SoOs of the VPN routes carrying at least one of the SoOs specified in the set with 100:1 and 200:1.</p>

Verifying the Route Attribute Set Configuration

After configuring route attribute sets, verify the configuration.

Prerequisites

Route attribute sets have been configured.

Procedure

- Run the **display xpl { as-path-list | community-list | ext-community-list rt | ext-community-list soo | ip-prefix-list | ipv6-prefix-list | route-filter | rd-list | route-flow-group | interface-list } [name xpl-name attachpoints]** command to check the configurations of XPL route attribute sets or the detailed information referenced by routing protocols.
- Run the **display xpl { as-path-list | community-list | ext-community-list rt | ext-community-list soo | ip-prefix-list | ipv6-prefix-list | rd-list | route-filter | route-flow-group | interface-list } state [attached | unattached]**

command to check information about XPL route attribute set configurations and references.

 **NOTE**

To check detailed XPL configurations, run the **display current-configuration configuration xpl** command.

----End

1.1.12.2.6 Configuring a Route-Filter

A route-filter can use its condition clause to filter routes and use its action clause to set or modify route attributes of the routes that match the condition clause. This section describes how to configure route-filters.

Usage Scenario

Route-filters include various matching rules and can flexibly apply to a variety of scenarios. Route-filters consist of condition clauses and action clauses, use condition clauses to filter routes based on sets or a single element, and use action clauses to modify route attributes of the routes that meet matching rules.

- Condition clause: A condition clause is defined based on a set or single element, beginning with an introductory condition clause in most cases. The action specified in the action clause is applied only to the routes that match the conditions specified in the condition clause.
- Action clause: An action clause specifies an action to be applied to the routes that match the conditions specified in the condition clause. An action clause determines whether the routes match the route-filter or modifies their route attributes.

Pre-configuration Tasks

Before configuring route-filters, complete the following tasks:

- Determine whether to configure **route attribute sets**.
- Configure a global variable set if some values need to be frequently used.

Procedure

- Configure a route-filter using the paragraph editing mode.
 - a. Run the **edit xpl route-filter route-filter-name** command to enter the route-filter paragraph editing interface view.
 - b. Press **i** to enter the text editing mode.

 **NOTE**

Sets or route-filters can be configured only in the text editing mode. If you exit from the text editing mode, you can perform shortcut key operations only.

- c. Configure a start clause in the format of **xpl route-filter route-filter-name(\$var1,\$var2,...)** for a route-filter. The parameters are optional and can be used in condition or action clauses.
- d. Configure a matching condition in the format of **if+condition clause +then** and connect the conditions with the Boolean operator **NOT**, **AND**, or **OR**. For details about condition clauses, see **Condition Clauses**.

 NOTE

Route-filters can have only action clauses and can also be empty (configured with only a start clause and an end clause). In this case, the default action **refuse** is used. If an empty route-filter is specified in another route-filter using a **call** clause, the empty route-filter does not take effect.

- e. Configure an action clause. For details about action clauses, see [Action Clauses](#).

 NOTE

- Multiple action clauses can be configured if they do not conflict with each other.
- Action clauses (excluding **approve**, **refuse**, **finish**, **break**, and **call route-filter route-filter-name**) must follow **apply**.

- f. (Optional) Configure **elseif+condition clause+then** to filter the routes that fail to match the conditions specified in the **if** clause and specify an action clause for the **elseif** clause. You can configure multiple **elseif** clauses to filter the routes that fail to meet the previous matching rule or configure an **else** clause to match the routes that fail to meet all the previous matching rules. Each **if**, **elseif**, or **else** clause must be followed by an action clause.
- g. Configure a conclusive condition clause (**endif**).

 NOTE

Steps 2 to 5 describe how to configure an **if** condition branch. One route-filter can have multiple **if** condition branches, and the **if** condition branches can be configured as follows:

- One **if** condition branch is followed by another.
- The **if+condition clause+then** or **elseif+condition clause+then** is followed by another **if** condition branch. Such a configuration further filters routes that match **if+condition clause+then** or **elseif+condition clause+then** against the second **if** condition branch.

Regardless of the configuration mode, route filtering continues until **finish**, **break**, **refuse**, or the last **if** condition branch.

- h. Configure a conclusive clause (**end-filter**) for the route-filter.
- i. Press **Esc** to exit from the text editing mode.
- j. Press **:wq** and **Enter** to save the configurations and exit from the global variable set paragraph-by-paragraph editing view.

 NOTE

A message is displayed for you to confirm whether to commit the configurations when you attempt to exit from the global variable set paragraph-by-paragraph editing view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph-by-paragraph editing view without saving the configurations, press **:q!** and **Enter**.

- Configure a route-filter using the line editing mode.
 - a. Run the **system-view** command to enter the system view.
 - b. Run the **xpl route-filter route-filter-name** command to enter the route-filter view.
 - c. Run the **if [not] condition-clause [{ and | or } [not] condition-clause] * then** command to configure matching rules for the route-filter.

 NOTE

Route-filters can have only action clauses and can also be empty (only configured with **xpl route-filter** and **end-filter**). In this case, the default action **refuse** is used. If an empty route-filter is specified in another route-filter using a **call** clause, the empty route-filter does not take effect.

- d. Configure an action using any of the following commands for the routes that meet matching rules:
 - **apply action-clause**: sets a route attribute for the routes that meet matching rules.
 - **approve**: further filters the routes that meet the matching rules of the current **if** condition branch against the next **if** condition branch.
 - **refuse**: denies routes that meet matching rules.
 - **finish**: completes route filtering and allows the routes that meet matching rules to match the route-filter.
 - **call route-filter route-filter-name**: further matches the routes that meet the matching rules of the current route-filter against the specified route-filter.
 - **break**: enables the device to exit from the current route-filter. If the current route-filter is referenced by a parent route-filter, the device keeps implementing remaining condition and action clauses of the parent route-filter.

 NOTE

Multiple action clauses can be configured if they do not conflict with each other.

- e. Run the **elseif [not] condition-clause [{ and | or } [not] condition-clause] * then** command to filter the routes that fail to meet the rules specified in the **if** command and specify an action for the routes. You can run the **elseif** command multiple times to filter the routes that fail to meet the previous matching rules or run the **else** command to match the routes that fail to meet all the previous matching rules.
- f. Run the **endif** command to conclude the configuration of the current **if** condition branch.

 NOTE

An **if** condition branch begins with **if** and ends with **endif**. One route-filter can have multiple **if** condition branches, and the **if** condition branches can be configured as follows:

- One **if** condition branch is followed by another.
- The **if** or **elseif** clause of one **if** condition branch is followed by another **if** condition branch. Such a configuration further filters routes that match **if** or **elseif** clause against the second **if** condition branch.

Regardless of the configuration mode, route filtering continues until **finish**, **break**, **refuse**, or the last **if** condition branch.

- g. Run the **end-filter** command to conclude the configuration of the route-filter.

h. Run the **commit** command to commit the configuration.

----End

Example

NOTE

If only paragraph-by-paragraph editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph-by-paragraph editing.

For detailed set and route-filter configuration steps, see [Configuration Procedures for Sets and Route-Filters Using the Paragraph-by-Paragraph Editing Mode](#). For details about XPL clauses, see [1.1.12.2.9 XPL Paragraph Editing Clauses](#).

Objective	Configure a route-filter to set the priorities of the BGP routes destined for 1.1.1.1 to 200.
Configuration Example 1	<pre><HUAWEI> edit xpl route-filter r1 xpl route-filter r1 if ip route-destination in { 1.1.1.1 32 } then apply preference 200 endif end-filter</pre> <p>The IPv4 prefix set used by the preceding route-filter includes only one element. In this situation, use the format of {element A, element B...} to present the IPv4 prefix set.</p>
Objective	Create a route-filter with a pre-defined variable and configure another route-filter to reference the preceding route-filter.
Configuration Example 2	<pre><HUAWEI> edit xpl route-filter para xpl route-filter para(\$var) apply med \$var end-filter <HUAWEI> edit xpl route-filter r2 xpl route-filter r2 if ip route-destination in aaa then call route-filter para(20) elseif ip route-destination in bbb then call route-filter para(30) endif end-filter</pre> <p>aaa and bbb specified in this route-filter are IPv4 prefix sets, and a maximum of eight parameters can be referenced by each route-filter.</p>

Objective	Configure a route-filter to set the priorities of the BGP routes carrying destination IP address 1.1.1.1 and MPLS labels to 50.
Configuration Example 3	<pre><HUAWEI> edit xpl route-filter r3 xpl route-filter r3 if ip route-destination in { 1.1.1.1 32 } and mpls- label exist then apply preference 50 endif end-filter</pre> <p>The conditions in a condition clause can be connected with the Boolean operator NOT, AND, or OR.</p>

Checking the Configurations

After configuring route-filters, check the configurations.

- Run the **display xpl route-filter [name *xpl-name* { attachpoints | uses | detail }]** command to check the XPL route-filter configurations or the detailed information referenced by routing protocols.
- Run the **display xpl route-filter state [attached | unattached]** command to check information about XPL route-filter configurations and references.
- Run the **display xpl statistics** command to check statistics about XPL and route-filter configurations.
- (Optional) Run the **reset xpl-filter *filter-name* counters** command to clear statistics about the routes matching the specified route-filter.

1.1.12.2.7 Applying Route-Filters

After a route-filter is created, it must be referenced by a route-policy to take effect.

Context

Currently, route-filters apply to OSPF/OSPFv3, RIP/RIPng, IS-IS, BGP/BGP4+. Perform one or more of the following configurations as required:

- [Apply route-filters to OSPF.](#)
- [Apply route-filters to RIP.](#)
- [Apply route-filters to RIPng.](#)
- [Apply route-filters to IS-IS.](#)
- [Apply route-filters to BGP.](#)
- [Apply route-filters to BGP4+.](#)

Procedure

- Apply route-filters to OSPF.

Route-filters can be used in OSPF route selection and interaction between OSPF and other routing protocols. [Table 1-192](#) lists usage scenarios and relevant configuration tasks.

Table 1-192 Route-filter application in OSPF

Usage Scenario	Relevant Configuration Task
Use a route-filter to filter the routes to be imported by OSPF.	Configuring OSPF to Import External Routes
Use a route-filter to filter the default routes to be imported by OSPF.	Configuring OSPF to Import a Default Route
Use a route-filter to filter LSAs in an Area.	Configuring OSPF to Filter LSAs in an Area
Use a route-filter to filter the routes received by OSPF.	Configuring OSPF to Filter Received Routes

- Apply route-filters to RIP.

Route-filters can be used in RIP route selection and interaction between RIP and other routing protocols. [Table 1-193](#) lists usage scenarios and relevant configuration tasks.

Table 1-193 Route-filter application in RIP

Usage Scenario	Relevant Configuration Task
Use a route-filter to filter the routes to be imported by RIP.	Configuring RIP to Import External Routes
Use a route-filter to filter the default routes to be advertised by RIP.	Configuring RIP to Advertise Default Routes

- Apply route-filters to RIPng.

Route-filters can be used in interaction between RIPng and other routing protocols. [Table 1-194](#) lists usage scenarios and relevant configuration tasks.

Table 1-194 Route-filter application in RIPng

Usage Scenario	Relevant Configuration Task
Use a route-filter to filter the routes to be imported by RIPng.	Configuring RIPng to Import External Routes

- Apply route-filters to IS-IS.

Route-filters can be used in IS-IS route selection and interaction between IS-IS and other routing protocols. **Table 1-195** lists usage scenarios and relevant configuration tasks.

Table 1-195 Route-filter application in IS-IS

Usage Scenario	Relevant Configuration Task
Use a route-filter to control IS-IS route leaking.	Configure IPv4 IS-IS route leaking Configure IPv6 IS-IS route leaking
Use a route-filter to control IS-IS default route generation.	Configure IS-IS to generate default IPv4 routes Configure IS-IS to generate default IPv6 routes
Use a route-filter to filter the routes to be imported by IS-IS.	Configure IS-IS to import external IPv4 routes Configure IS-IS to import external IPv6 routes

- Apply route-filters to BGP.

Route-filters can be used to set BGP route attributes and filter the routes to be imported, advertised, or received. **Table 1-196** lists usage scenarios and relevant configuration tasks.

Table 1-196 Route-filter application in BGP

Usage Scenario	Relevant Configuration Task
Use a route-filter to set a priority for BGP routes.	Setting the BGP Preference
Use a route-filter to filter the routes to be imported by BGP.	Configuring BGP to Import Routes
Use a route-filter to filter the routes to be advertised by BGP.	Using XPL to Filter the BGP Routes to Be Advertised
Use a route-filter to filter the routes to be received by BGP.	Using XPL to Filter the BGP Routes to Be Received

- Apply route-filters to BGP4+.

Route-filters can be used to set BGP4+ route attributes and filter the routes to be imported, advertised, or received. **Table 1-197** lists usage scenarios and relevant configuration tasks.

Table 1-197 Route-filter application in BGP4+

Usage Scenario	Relevant Configuration Task
Use a route-filter to set a priority for BGP4+ routes.	Configuring BGP4+ to Import Routes
Use a route-filter to filter the routes to be imported by BGP4+.	Setting a BGP4+ Preference
Use a route-filter to filter the routes to be advertised by BGP4+.	Using XPL to Filter the BGP4+ Routes to Be Advertised
Use a route-filter to filter the routes to be received by BGP4+.	Using XPL to Filter the BGP4+ Routes to Be Received

----End

1.1.12.2.8 Configuration Examples for XPL

This section describes several XPL configuration examples.

Example for Using XPL to Filter the Routes to Be Accepted and Advertised (Paragraph-by-Paragraph Editing)

This section provides an example for using XPL to filter the routes to be accepted and advertised. This section provides an example for using XPL to filter the routes based on sets and route-filters in paragraph-by-paragraph editing mode.

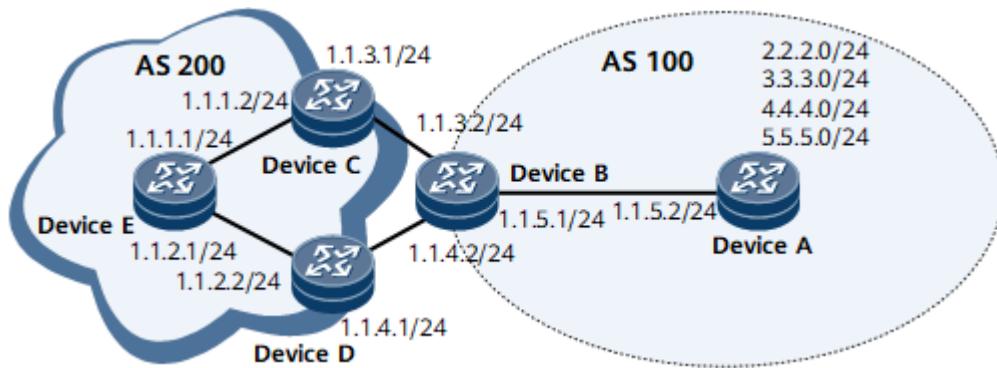
Networking Requirements

On the BGP network shown in [Figure 1-475](#), DeviceA and DeviceB belong to AS 100, and DeviceC, DeviceD, and DeviceE belong to AS 200. DeviceA imports routes 2.2.2.0/24, 3.3.3.0/24, 4.4.4.0/24, and 5.5.5.0/24. Requirements:

- DeviceA advertises routes 2.2.2.0/24, 3.3.3.0/24, and 4.4.4.0/24 only to DeviceB.
- After receiving the three routes, DeviceC needs to advertise all routes to DeviceE. DeviceD needs to advertise only routes 2.2.2.0/24 and 3.3.3.0/24 to DeviceE and the MED value of the route 2.2.2.0/24 must be greater than that of the route 2.2.2.0/24 advertised by DeviceC, so that DeviceE selects DeviceC as the egress for the traffic destined for 2.2.2.0/24.

To meet the preceding requirements, configure export or import policies. In this example, an export policy is configured on DeviceA, and two import policies are configured on DeviceE.

Figure 1-475 Network diagram of filtering routes to be advertised and accepted



Device Name	Interface	IP Address
DeviceA	GE 3/0/0	1.1.5.2/24
DeviceB	GE 3/0/0	1.1.5.1/24
DeviceB	GE 3/0/1	1.1.4.2/24
DeviceB	GE 3/0/2	1.1.3.2/24
DeviceC	GE 3/0/1	1.1.1.2/24
DeviceC	GE 3/0/2	1.1.3.1/24
DeviceD	GE 3/0/1	1.1.2.2/24
DeviceD	GE 3/0/2	1.1.4.1/24
DeviceE	GE 3/0/1	1.1.2.1/24
DeviceE	GE 3/0/2	1.1.1.1/24

Configuration Precautions

During the configuration, note the following:

- A prefix range needs to be specified for the prefix set as required.
- Prefix set names are case-sensitive.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic BGP functions on DeviceA, DeviceB, DeviceC, DeviceD, and DeviceE.
2. Configure static routes on DeviceA and import them to the BGP routing table.
3. Configure an export policy on DeviceA and check the filtering result on DeviceB.

4. Configure an import policy on DeviceE and check the filtering result on DeviceE.

Data Preparation

To complete the configuration, you need the following data:

- Five static routes imported on DeviceA
- DeviceA and DeviceB in AS 100, DeviceC, DeviceD, and DeviceE in AS 200.
- Name of an IPv4 prefix set and the routes to be filtered

Procedure

Step 1 Assign an IP address to each interface. For details, see [Configuration Files](#).

Step 2 Configure basic BGP functions.

Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 1.1.5.1 as-number 100
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] bgp 100
[*DeviceB-bgp] peer 1.1.5.2 as-number 100
[*DeviceB-bgp] peer 1.1.3.1 as-number 200
[*DeviceB-bgp] peer 1.1.4.1 as-number 200
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

Configure DeviceC.

```
<DeviceC> system-view
[~DeviceC] bgp 200
[*DeviceC-bgp] peer 1.1.1.1 as-number 200
[*DeviceC-bgp] peer 1.1.3.2 as-number 100
[*DeviceC-bgp] commit
[~DeviceC-bgp] quit
```

Configure DeviceD.

```
<DeviceD> system-view
[~DeviceD] bgp 200
[*DeviceD-bgp] peer 1.1.2.1 as-number 200
[*DeviceD-bgp] peer 1.1.4.2 as-number 100
[*DeviceD-bgp] commit
[~DeviceD-bgp] quit
```

Configure DeviceE.

```
<DeviceE> system-view
[~DeviceE] bgp 200
[*DeviceE-bgp] peer 1.1.1.2 as-number 200
[*DeviceE-bgp] peer 1.1.2.2 as-number 200
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

Step 3 Configure five static routes on DeviceA and import them to the BGP routing table.

```
[~DeviceA] ip route-static 2.2.2.0 255.255.255.0 NULL0
```

```
[*DeviceA] ip route-static 3.3.3.0 255.255.255.0 NULL0
[*DeviceA] ip route-static 4.4.4.0 255.255.255.0 NULL0
[*DeviceA] ip route-static 5.5.5.0 255.255.255.0 NULL0
[*DeviceA] bgp 100
[*DeviceA-bgp] import-route static
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

Check the BGP routing table of DeviceB. The command output shows that DeviceB has learned the imported static routes.

```
[~DeviceB] display bgp routing-table
```

```
BGP Local router ID is 1.1.5.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

```
Total Number of Routes: 4
Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
*>i  2.2.2.0/24    1.1.5.2      0        100     0       ?
*>i  3.3.3.0/24    1.1.5.2      0        100     0       ?
*>i  4.4.4.0/24    1.1.5.2      0        100     0       ?
*>i  5.5.5.0/24    1.1.5.2      0        100     0       ?
```

Check the BGP routing table on DeviceE. The command output shows that BGP has imported four static routes (one from DeviceC and one from DeviceD).

```
[~DeviceE] display bgp routing-table
```

```
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

```
Total Number of Routes: 8
Network          NextHop        MED      LocPrf  PrefVal Path/Ogn
i   2.2.2.0/24    1.1.3.2      100     0       100?
i   1.1.4.2       1.1.4.2      100     0       100?
i   3.3.3.0/24    1.1.4.2      100     0       100?
i   1.1.3.2       1.1.3.2      100     0       100?
i   4.4.4.0/24    1.1.3.2      100     0       100?
i   1.1.4.2       1.1.4.2      100     0       100?
i   5.5.5.0/24    1.1.4.2      100     0       100?
i   1.1.3.2       1.1.3.2      100     0       100?
```

Step 4 Configure a route advertisement policy on DeviceA.

Configure an IPv4 prefix set named **prefix1** on DeviceA.

```
[~DeviceA] quit
< Device A> edit xpl ip-prefix-list prefix1 //Enter the IPv4 prefix set paragraph-by-paragraph editing view.
xpl ip-prefix-list prefix1
2.2.2.0 24,
3.3.3.0 24,
4.4.4.0 24
end-list
```

 **NOTE**

After entering the paragraph-by-paragraph editing view, press **i** to enter the text editing mode.

After performing the configuration in the text editing mode of the paragraph-by-paragraph editing view, press **Esc** to exit from the text editing mode, and press :**wq** and **Enter** to save the configuration and exit from the paragraph-by-paragraph editing view. For details, see "Example of Paragraph-by-Paragraph Editing Operations" in [1.1.12.2.3 Introduction to XPL Paragraph-by-Paragraph Editing](#).

After entering the XPL paragraph-by-paragraph editing view, press **i**. -- **INSERT** -- is displayed at the bottom of the interface, indicating that you can enter content. Configure an IPv4 prefix set named **prefix1** in the paragraph-by-paragraph editing view and separate elements with commas (,), as shown in the following figure.

Press **Esc** to exit the text editing mode. -- **INSERT** -- at the bottom of the interface then disappears. Enter **:wq** and press **Enter** to save the configuration and exit the paragraph-by-paragraph editing view. The system asks you whether to commit the configuration. If you enter **y** and return to the user view, the configuration is saved successfully.

Proceed with commit (yes/no/cancel)? [Y/N/C]:y

<DeviceA>

To add or delete a prefix, run the **edit xpl ip-prefix-list *prefix1*** command again to enter the paragraph-by-paragraph editing view. Press **i**. After -- **INSERT** -- is displayed at the bottom of the interface, use arrow keys to move the cursor to the desired position and modify the prefix. Use commas (,) to separate elements. Otherwise, an error may occur, and the configuration cannot be committed. For example, to delete the prefix 2.2.2.0 24, use arrow keys to move the cursor to the corresponding position and delete the prefix, as shown in the following figure.

```
xpl ip-prefix-list prefix1  
 2.2/  
 3.3.3.0 24,  
 4.4.4.0 24  
end-list
```

After the modification is complete, press **Esc** to exit the text editing mode. --
INSERT -- at the bottom of the interface disappears. Enter :wq and press **Enter** to save the configuration and exit the paragraph-by-paragraph editing view. The system asks you whether to commit the configuration. If you enter **y** and return to the user view, the configuration is successfully modified. The following sections do not provide detailed XPL operations. Only the content entered on the paragraph-by-paragraph editing interface is shown.

```
<DeviceA> # Configure a route-filter named r1
routes 2.2.2.0/24, 3.3.3.0/24,
< Device A> edit xpl route-filter r1 /
xpl route-filter r1
if ip route-destination in prefix1 then
    approve
else
    refuse
endif
end-filter
```

```
# Configure a route-filter named r1 on Device A, apply prefix1 to r1, and permit routes 2.2.2.0/24, 3.3.3.0/24, and 4.4.4.0/24.
```

< Device A> **edit xpl route-filter r1** //Enter the route-filter paragraph-by-paragraph editing view.

```
xpl route-filter r1
if ip route-destination in prefix1 then
    approve
else
    refuse
endif
end-filter
```

```
# Configure an advertisement policy on DeviceA and use r1 to filter the routes to be advertised to DeviceB.
```

```
<DeviceA> system-view
[~DeviceA] bgp 100
[~DeviceA-bgp] peer 1.1.5.1 route-filter r1 export
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

Check the BGP routing table of DeviceB. The following command output shows that the route 5.5.5.0/24 is not included in the BGP routing table of DeviceB.

[~DeviceB] display bgp routing-table

BGP Local router ID is 1.1.5.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
h - history, i - internal, s - suppressed, S - Stale
Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found

Total Number of Routes: 3						
	Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
*>i	2.2.2.0/24	1.1.5.2	0	100	0	?
*>i	3.3.3.0/24	1.1.5.2	0	100	0	?
*>i	4.4.4.0/24	1.1.5.2	0	100	0	?

Step 5 Configure a policy for receiving routes on DeviceE.

```
# Configure a route-filter named appmed with parameters on DeviceE to set MEDs for routes.
```

```
[~DeviceE] quit  
< Device E> edit xpl route-filter appmed //Enter the route-filter paragraph-by-paragraph editing view.  
  
xpl route-filter appmed($med)  
apply med $med  
end-filter
```

```
# Configure route-filter named r2 on DeviceE to permit routes 2.2.2.0/24 and  
3.3.3.0/24 and use appmed to set the MED of the route 2.2.2.0/24 to 200.
```

< Device E> **edit xpl route-filter r2** //Enter the route-filter paragraph-by-paragraph editing view.

```
xpl route-filter r2
if ip route-destination in {2.2.2.0 24} then
call route-filter appmed(200)
elseif ip route-destination in {2.2.2.0 24, 3.3.3.0 24} then
approve
else
refuse
endif
end-filter
```

```
# Configure a route-filter named r3 on Device E to set the MED of 2.2.2.0/24 to 100.
```

```
< Device E> edit xpl route-filter r3 //Enter the route-filter paragraph-by-paragraph editing view.  
xpl route-filter r3  
if ip route-destination in {2.2.2.0 24} then  
call route-filter appmed(100)  
else  
approve  
endif  
end-filter
```

Configure an import policy on DeviceE, and apply **r2** to the export policy for the routes advertised by DeviceD and **r3** to the export policy for the routes advertised by DeviceC.

```
<DeviceE> system-view
[~DeviceE] bgp 200
[~DeviceE-bgp] peer 1.1.2.2 route-filter r2 import
[*DeviceE-bgp] peer 1.1.1.2 route-filter r3 import
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

Check the BGP routing table of DeviceE. The following command output shows that the route 4.4.4.0/24 learned from DeviceD is not included in the BGP routing table of DeviceE, and the MED values of the routes 2.2.2.0/24 learned from DeviceC and DeviceD are 100 and 200, respectively.

```
[~DeviceE] display bgp routing-table
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 5						
	Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
i	2.2.2.0/24	1.1.4.2	200	100	0	100?
i		1.1.3.2	100	100	0	100?
i	3.3.3.0/24	1.1.4.2		100	0	100?
i		1.1.3.2		100	0	100?
i	4.4.4.0/24	1.1.3.2		100	0	100?

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 1.1.5.2 255.255.255.0  
#  
bgp 100  
peer 1.1.5.1 as-number 100  
#  
ipv4-family unicast  
undo synchronization  
import-route static  
peer 1.1.5.1 enable  
peer 1.1.5.1 route-filter r1 export  
#  
ip route-static 2.2.2.0 255.255.255.0 NULL0  
ip route-static 3.3.3.0 255.255.255.0 NULL0  
ip route-static 4.4.4.0 255.255.255.0 NULL0  
ip route-static 5.5.5.0 255.255.255.0 NULL0  
#  
xpl route-filter r1  
if ip route-destination in prefix1 then  
    approve  
else  
    refuse  
endif  
end-filter  
#
```

```
xpl ip-prefix-list prefix1
2.2.2.0 24,
3.3.3.0 24,
4.4.4.0 24
end-list
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet3/0/0
undo shutdown
ip address 1.1.5.1 255.255.255.0
#
interface GigabitEthernet3/0/1
undo shutdown
ip address 1.1.4.2 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 1.1.3.2 255.255.255.0
#
bgp 100
peer 1.1.3.1 as-number 200
peer 1.1.4.1 as-number 200
peer 1.1.5.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.3.1 enable
peer 1.1.4.1 enable
peer 1.1.5.2 enable
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet3/0/1
undo shutdown
ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 1.1.3.1 255.255.255.0
#
bgp 200
peer 1.1.1.1 as-number 200
peer 1.1.3.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.1.1 enable
peer 1.1.3.2 enable
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet3/0/1
undo shutdown
ip address 1.1.2.2 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
```

```
ip address 1.1.4.1 255.255.255.0
#
bgp 200
peer 1.1.2.1 as-number 200
peer 1.1.4.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.2.1 enable
peer 1.1.4.2 enable
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet3/0/1
undo shutdown
ip address 1.1.2.1 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 1.1.1.1 255.255.255.0
#
bgp 200
peer 1.1.1.2 as-number 200
peer 1.1.2.2 as-number 200
#
ipv4-family unicast
undo synchronization
peer 1.1.1.2 enable
peer 1.1.1.2 route-filter r3 import
peer 1.1.2.2 enable
peer 1.1.2.2 route-filter r2 import
#
xpl route-filter appmed($med)
apply med $med
end-filter
#
xpl route-filter r2
if ip route-destination in {2.2.2.0 24} then
call route-filter appmed(200)
elseif ip route-destination in {2.2.2.0 24, 3.3.3.0 24} then
approve
else
refuse
endif
end-filter
#
xpl route-filter r3
if ip route-destination in {2.2.2.0 24} then
call route-filter appmed(100)
else
approve
endif
end-filter
#
return
```

Example for Using XPL to Filter the Routes to Be Received and Advertised (Line-by-Line Editing)

This section provides an example for using XPL to filter the routes to be received and advertised based on sets and route-filters in line editing mode.

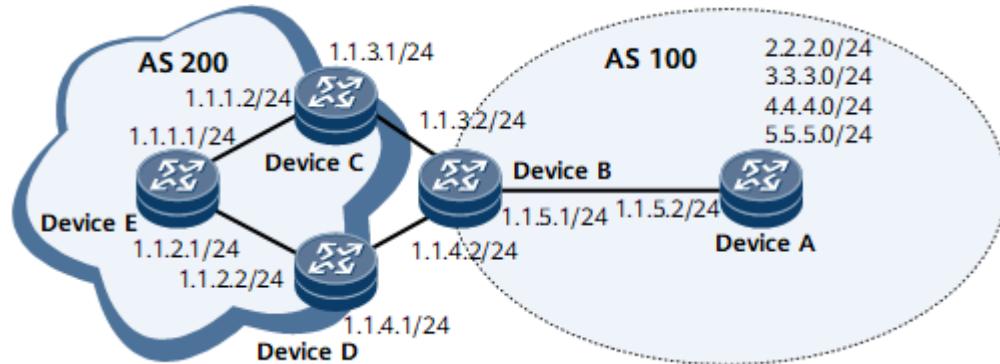
Networking Requirements

On the BGP network shown in [Figure 1-476](#), DeviceA and DeviceB belong to AS 100, and DeviceC, DeviceD, and DeviceE belong to AS 200. DeviceA imports routes 2.2.2.0/24, 3.3.3.0/24, 4.4.4.0/24, and 5.5.5.0/24. Requirements:

- DeviceA advertises routes 2.2.2.0/24, 3.3.3.0/24, and 4.4.4.0/24 only to DeviceB.
- After receiving the three routes, DeviceC needs to advertise all routes to DeviceE. DeviceD needs to advertise only routes 2.2.2.0/24 and 3.3.3.0/24 to DeviceE and the MED value of the route 2.2.2.0/24 must be greater than that of the route 2.2.2.0/24 advertised by DeviceC, so that DeviceE selects DeviceC as the egress for the traffic destined for 2.2.2.0/24.

To meet the preceding requirements, configure export or import policies. In this example, an export policy is configured on DeviceA, and two import policies are configured on DeviceE.

Figure 1-476 Network diagram of filtering routes to be advertised and accepted



Device Name	Interface	IP Address
DeviceA	GE 3/0/0	1.1.5.2/24
DeviceB	GE 3/0/0	1.1.5.1/24
DeviceB	GE 3/0/1	1.1.4.2/24
DeviceB	GE 3/0/2	1.1.3.2/24
DeviceC	GE 3/0/1	1.1.1.2/24
DeviceC	GE 3/0/2	1.1.3.1/24
DeviceD	GE 3/0/1	1.1.2.2/24
DeviceD	GE 3/0/2	1.1.4.1/24
DeviceE	GE 3/0/1	1.1.2.1/24
DeviceE	GE 3/0/2	1.1.1.1/24

Configuration Precautions

During the configuration, note the following:

- A prefix range needs to be specified for the prefix set as required.
- Prefix set names are case-sensitive.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic BGP functions on Device A, Device B, Device C, Device D, and Device E.
2. Configure four static routes on Device A and import them into BGP.
3. Configure an export policy on Device A and check the BGP routing table of Device B.
4. Configure two import policies on Device E and check the BGP routing table of Device E.

Data Preparation

To complete the configuration, you need the following data:

- Four static routes to be imported by Device A
- AS 100 (for Device A and Device B) and AS 200 (Device C, Device D, and Device E)
- Name of an IPv4 prefix set and the routes to be filtered

Procedure

Step 1 Assign an IP address to each interface. For configuration details, see [Configuration Files](#) in this section.

Step 2 Configure basic BGP functions.

Configure Device A.

```
<DeviceA> system-view
[~DeviceA] bgp 100
[*DeviceA-bgp] peer 1.1.5.1 as-number 100
[*DeviceA-bgp] commit
[~DeviceA-bgp] quit
```

Configure Device B.

```
<DeviceB> system-view
[~DeviceB] bgp 100
[*DeviceB-bgp] peer 1.1.5.2 as-number 100
[*DeviceB-bgp] peer 1.1.3.1 as-number 200
[*DeviceB-bgp] peer 1.1.4.1 as-number 200
[*DeviceB-bgp] commit
[~DeviceB-bgp] quit
```

Configure Device C:

```
<DeviceC> system-view
[~DeviceC] bgp 200
[*DeviceC-bgp] peer 1.1.1.1 as-number 200
[*DeviceC-bgp] peer 1.1.3.2 as-number 100
```

```
[*DeviceC-bgp] commit  
[~DeviceC-bgp] quit
```

Configure Device D:

```
<DeviceD> system-view  
[~DeviceD] bgp 200  
[*DeviceD-bgp] peer 1.1.2.1 as-number 200  
[*DeviceD-bgp] peer 1.1.4.2 as-number 100  
[*DeviceD-bgp] commit  
[~DeviceD-bgp] quit
```

Configure Device E.

```
<DeviceE> system-view  
[~DeviceE] bgp 200  
[*DeviceE-bgp] peer 1.1.1.2 as-number 200  
[*DeviceE-bgp] peer 1.1.2.2 as-number 200  
[*DeviceE-bgp] commit  
[~DeviceE-bgp] quit
```

Step 3 Configure four static routes on Device A and import them into BGP.

```
[~DeviceA] ip route-static 2.2.2.0 255.255.255.0 NULL0  
[*DeviceA] ip route-static 3.3.3.0 255.255.255.0 NULL0  
[*DeviceA] ip route-static 4.4.4.0 255.255.255.0 NULL0  
[*DeviceA] ip route-static 5.5.5.0 255.255.255.0 NULL0  
[*DeviceA] bgp 100  
[*DeviceA-bgp] import-route static  
[*DeviceA-bgp] commit  
[~DeviceA-bgp] quit
```

Check the BGP routing table of Device B. The following command output shows that the four static routes have been added to the BGP routing table of Device B.

```
[~DeviceB] display bgp routing-table
```

```
BGP Local router ID is 1.1.5.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 4					
	Network	NextHop	MED	LocPrf	PrefVal Path/Ogn
*>i	2.2.2.0/24	1.1.5.2	0	100	0 ?
*>i	3.3.3.0/24	1.1.5.2	0	100	0 ?
*>i	4.4.4.0/24	1.1.5.2	0	100	0 ?
*>i	5.5.5.0/24	1.1.5.2	0	100	0 ?

Check the BGP routing table of Device E. The following command output shows that two copies of the four static routes from Device C and Device D have been added to the BGP routing table of Device E.

```
[~DeviceE] display bgp routing-table
```

```
BGP Local router ID is 1.1.1.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 8					
	Network	NextHop	MED	LocPrf	PrefVal Path/Ogn
i	2.2.2.0/24	1.1.3.2	100	0	100?
i		1.1.4.2	100	0	100?

```
i 3.3.3.0/24      1.1.4.2          100   0   100?  
i           1.1.3.2          100   0   100?  
i 4.4.4.0/24      1.1.3.2          100   0   100?  
i           1.1.4.2          100   0   100?  
i 5.5.5.0/24      1.1.4.2          100   0   100?  
i           1.1.3.2          100   0   100?
```

Step 4 Configure an export policy on Device A.

```
# Configure an IPv4 prefix set named prefix1 on Device A.
```

```
[~DeviceA] xpl ip-prefix-list prefix1  
[~DeviceA-xpl-pfx] 2.2.2.0 24,  
[~DeviceA-xpl-pfx] 3.3.3.0 24,  
[~DeviceA-xpl-pfx] 4.4.4.0 24  
[~DeviceA-xpl-pfx] end-list  
[*DeviceA] commit
```

```
# Configure a route-filter named r1 on Device A, apply prefix1 to r1, and permit routes 2.2.2.0/24, 3.3.3.0/24, and 4.4.4.0/24.
```

```
[~DeviceA] xpl route-filter r1  
[~DeviceA-xpl-filter] if ip route-destination in prefix1 then  
[~DeviceA-xpl-filter-if] approve  
[~DeviceA-xpl-filter-if] else  
[~DeviceA-xpl-filter-else] refuse  
[~DeviceA-xpl-filter-else] endif  
[~DeviceA-xpl-filter] end-filter  
[*DeviceA] commit
```

```
# Configure an export policy on Device A and apply r1 to the policy to filter the routes to be advertised to Device B.
```

```
<DeviceA> system-view  
[~DeviceA] bgp 100  
[~DeviceA-bgp] peer 1.1.5.1 route-filter r1 export  
[*DeviceA-bgp] commit  
[~DeviceA-bgp] quit
```

```
# Check the BGP routing table of Device B. The following command output shows that the route 5.5.5.0/24 is not included in the BGP routing table of Device B.
```

```
[~DeviceB] display bgp routing-table  
  
BGP Local router ID is 1.1.5.1  
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,  
             h - history, i - internal, s - suppressed, S - Stale  
Origin : i - IGP, e - EGP, ? - incomplete  
RPKI validation codes: V - valid, I - invalid, N - not-found  
  
Total Number of Routes: 3  
      Network        NextHop       MED     LocPrf  PrefVal Path/Ogn  
*>i 2.2.2.0/24    1.1.5.2      0       100      0      ?  
*>i 3.3.3.0/24    1.1.5.2      0       100      0      ?  
*>i 4.4.4.0/24    1.1.5.2      0       100      0      ?
```

Step 5 Configure two import policies on Device E.

```
# Configure a route-filter named appmed with a pre-defined variable to set the MED.
```

```
[~DeviceE] xpl route-filter appmed($med)  
[~DeviceE-xpl-filter] apply med $med  
[~DeviceE-xpl-filter] end-filter  
[*DeviceE] commit
```

Configure a route-filter named **r2** on Device E to permit routes 2.2.2.0/24 and 3.3.3.0/24 and use **appmed** to set the MED of the route 2.2.2.0/24 to 200.

```
[~DeviceE] xpl route-filter r2
[~DeviceE-xpl-filter] if ip route-destination in {2.2.2.0 24} then
[~DeviceE-xpl-filter-if] call route-filter appmed(200)
[~DeviceE-xpl-filter-if] elseif ip route-destination in {2.2.2.0 24, 3.3.3.0 24} then
[~DeviceE-xpl-filter-elif] approve
[~DeviceE-xpl-filter-elif] else
[~DeviceE-xpl-filter-else] refuse
[~DeviceE-xpl-filter-else] endif
[~DeviceE-xpl-filter] end-filter
[*DeviceE] commit
```

Configure a route-filter named **r3** on Device E to set the MED of the route 2.2.2.0/24 to 100.

```
[~DeviceE] xpl route-filter r3
[~DeviceE-xpl-filter] if ip route-destination in {2.2.2.0 24} then
[~DeviceE-xpl-filter-if] call route-filter appmed(100)
[~DeviceE-xpl-filter-if] else
[~DeviceE-xpl-filter-else] approve
[~DeviceE-xpl-filter-else] endif
[~DeviceE-xpl-filter] end-filter
[*DeviceE] commit
```

Configure an import policy on Device E for the routes learned from Device D and apply **r2** to the policy. In addition, configure another import policy on Device E for the routes learned from Device C and apply **r3** to the policy.

```
<DeviceE> system-view
[~DeviceE] bgp 200
[~DeviceE-bgp] peer 1.1.2.2 route-filter r2 import
[*DeviceE-bgp] peer 1.1.1.2 route-filter r3 import
[*DeviceE-bgp] commit
[~DeviceE-bgp] quit
```

Check the BGP routing table of Device E. The following command output shows that the route 4.4.4.0/24 learned from Device D is not included in the BGP routing table of Device B and that the MED values of the routes 2.2.2.0/24 learned from Device C and Device D are 100 and 200, respectively.

```
[~DeviceE] display bgp routing-table
BGP Local router ID is 1.1.1.1
Status codes: * - valid, > - best, d - damped, x - best external, a - add path,
              h - history, i - internal, s - suppressed, S - Stale
              Origin : i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V - valid, I - invalid, N - not-found
```

Total Number of Routes: 5						
	Network	NextHop	MED	LocPrf	PrefVal	Path/Ogn
i	2.2.2.0/24	1.1.4.2	200	100	0	100?
i		1.1.3.2	100	100	0	100?
i	3.3.3.0/24	1.1.4.2		100	0	100?
i		1.1.3.2		100	0	100?
i	4.4.4.0/24	1.1.3.2		100	0	100?

----End

Configuration Files

- DeviceA configuration file

```
#  
sysname DeviceA
```

```
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 1.1.5.2 255.255.255.0  
#  
bgp 100  
peer 1.1.5.1 as-number 100  
#  
ipv4-family unicast  
undo synchronization  
import-route static  
peer 1.1.5.1 enable  
peer 1.1.5.1 route-filter r1 export  
#  
ip route-static 2.2.2.0 255.255.255.0 NULL0  
ip route-static 3.3.3.0 255.255.255.0 NULL0  
ip route-static 4.4.4.0 255.255.255.0 NULL0  
ip route-static 5.5.5.0 255.255.255.0 NULL0  
#  
xpl route-filter r1  
if ip route-destination in prefix1 then  
    approve  
else  
    refuse  
endif  
end-filter  
#  
xpl ip-prefix-list prefix1  
2.2.2.0 24,  
3.3.3.0 24,  
4.4.4.0 24  
end-list  
#  
return
```

- DeviceB configuration file

```
#  
sysname DeviceB  
#  
interface GigabitEthernet3/0/0  
undo shutdown  
ip address 1.1.5.1 255.255.255.0  
#  
interface GigabitEthernet3/0/1  
undo shutdown  
ip address 1.1.4.2 255.255.255.0  
#  
interface GigabitEthernet3/0/2  
undo shutdown  
ip address 1.1.3.2 255.255.255.0  
#  
bgp 100  
peer 1.1.3.1 as-number 200  
peer 1.1.4.1 as-number 200  
peer 1.1.5.2 as-number 100  
#  
ipv4-family unicast  
undo synchronization  
peer 1.1.3.1 enable  
peer 1.1.4.1 enable  
peer 1.1.5.2 enable  
#  
return
```

- DeviceC configuration file

```
#  
sysname DeviceC  
#  
interface GigabitEthernet3/0/1  
undo shutdown
```

```
ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 1.1.3.1 255.255.255.0
#
bgp 200
peer 1.1.1.1 as-number 200
peer 1.1.3.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.1.1 enable
peer 1.1.3.2 enable
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet3/0/1
undo shutdown
ip address 1.1.2.2 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 1.1.4.1 255.255.255.0
#
bgp 200
peer 1.1.2.1 as-number 200
peer 1.1.4.2 as-number 100
#
ipv4-family unicast
undo synchronization
peer 1.1.2.1 enable
peer 1.1.4.2 enable
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet3/0/1
undo shutdown
ip address 1.1.2.1 255.255.255.0
#
interface GigabitEthernet3/0/2
undo shutdown
ip address 1.1.1.1 255.255.255.0
#
bgp 200
peer 1.1.1.2 as-number 200
peer 1.1.2.2 as-number 200
#
ipv4-family unicast
undo synchronization
peer 1.1.1.2 enable
peer 1.1.1.2 route-filter r3 import
peer 1.1.2.2 enable
peer 1.1.2.2 route-filter r2 import
#
xpl route-filter appmed($med)
apply med $med
end-filter
#
xpl route-filter r2
if ip route-destination in {2.2.2.0 24} then
call route-filter appmed(200)
```

```
elseif ip route-destination in {2.2.2.0 24, 3.3.3.0 24} then
    approve
else
    refuse
endif
end-filter
#
xpl route-filter r3
if ip route-destination in {2.2.2.0 24} then
    call route-filter appmed(100)
else
    approve
endif
end-filter
#
return
```

1.1.12.2.9 XPL Paragraph Editing Clauses

XPL paragraph editing clauses are configured in the XPL paragraph editing interface view and combined based on specified rules. XPL paragraph editing clauses are classified as follows:

Common Clauses

Common clauses are classified as logic, start, or end clauses. Logic clauses indicate logical relationships, the start clause begins a configuration in XPL, and the end clause concludes a configuration in XPL.

Logic Clauses

eq: equal to.

ge: greater than or equal to.

le: less than or equal to.

in: included in a set.

if: introductory condition clause.

elseif: introductory condition clause, used to filter the routes that fail to meet previous matching rules.

else: introductory condition clause, used to include all the routes that fail to meet previous matching rules.

then: introductory action clause, in the format of **if+condition clause+then** or **elseif+condition clause+then**. **else** is not followed by an introductory action clause in most cases.

apply: action implementation clause, in the format of **apply+action clause**. Action clauses (excluding **approve**, **refuse**, **finish**, **call route-filter route-filter-name**, and **break**) must follow **apply**.

endif: conclusive condition clause, following a condition and an action, and concluding all matching rules.

Configuration Start and End Clauses

Configuration Start Clauses	Configuration End Clauses
xpl global-value : begins the edition of a global variable set.	end-global-value : concludes the edition of a global variable set.
xpl ip-prefix-list <i>ip-prefix-list-name</i> : begins the edition of an IPv4 prefix set.	end-list : concludes the edition of an IPv4 prefix set.
xpl ipv6-prefix-list <i>ipv6-prefix-list-name</i> : begins the edition of an IPv6 prefix set.	end-list : concludes the edition of an IPv6 prefix set.
xpl as-path-list <i>as-path-list-name</i> : begins the edition of an AS_Path set.	end-list : concludes the edition of an AS_Path set.
xpl community-list <i>community-list-name</i> : begins the edition of a community set.	end-list : concludes the edition of a community set.
xpl large-community-list <i>large-community-list-name</i> : starts the edition of a Large-community set.	end-list : concludes the edition of a Large-community set.
xpl rd-list <i>rd-list-name</i> : begins the edition of an RD set.	end-list : concludes the edition of an RD set.
xpl extcommunity-list rt <i>rt-list-name</i> : begins the edition of a route target set.	end-list : concludes the edition of a route target set.
xpl extcommunity-list soo <i>soo-list-name</i> : begins the edition of a site of origin (SoO) set.	end-list : concludes the edition of an SoO set.
xpl route-filter <i>route-filter-name</i> : begins the edition of a route-filter.	end-filter : concludes the edition of a route-filter.
xpl route-flow-group <i>group-name</i> : begins the edition of a QPPB configuration group.	end-group : concludes the edition of a QPPB configuration group.

NOTE

A configuration end clause concludes only the edition of a set or route-filter in the paragraph editing interface view. To save the current configuration and exit this view, press **Ctrl+X**.

Condition Clauses

Condition clauses follow **if** or **elseif** and are used as matching rules to filter routes.

 NOTE

The parameters in condition clauses can be specific values or global variables referenced using \$+global variable name. The parameters in a route-filter with pre-defined variables can be those defined in this route-filter.

Global variables cannot be referenced by the route attribute sets in the format of {element A, element B...}.

Condition Clauses That Are Based on the Route Cost

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
med eq med	Matches the BGP routes with the specified <i>med</i> .	<i>med</i> : indicates an integer ranging from 0 to 429496 7295.	<ul style="list-style-type: none">● peer route-filter import● peer route-filter export● preference (BGP)● aggregate suppress-filter● aggregate origin-filter● dampening● import route-filter● export route-filter● qos-local-id● ingress-lsp trigger● routing-table rib-only● import-rib (BGP-VPN instance address family view)● import-rib vpn-instance (BGP public network address family view)● ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf● ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3● import-route (IS-IS)● default-route-advertise (IS-IS)● ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis● preference (IS-IS)● frr-policy route (IS-IS)● filter-policy export (IS-IS)● filter-policy import (IS-IS)● import-route isis level-1 into level-2 (IS-IS)● import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
med ge med	Matches the BGP routes with the MED greater than or equal to the specified <i>med</i> .	<i>med</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export ● preference (BGP) ● aggregate suppress-filter ● aggregate origin-filter ● dampening ● import route-filter ● export route-filter ● qos-local-id ● ingress-lsp trigger ● routing-table rib-only ● import-rib (BGP-VPN instance address family view) ● import-rib vpn-instance (BGP public network address family view) ● ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf ● ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 ● import-route (IS-IS) ● default-route-advertise (IS-IS) ● ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis ● preference (IS-IS) ● frr-policy route (IS-IS) ● filter-policy export (IS-IS) ● filter-policy import (IS-IS) ● import-route isis level-1 into level-2 (IS-IS) ● import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
med le med	Matches the BGP routes with the MED less than or equal to the specified <i>med</i> .	<i>med</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export ● preference (BGP) ● aggregate suppress-filter ● aggregate origin-filter ● dampening ● import route-filter ● export route-filter ● qos-local-id ● ingress-lsp trigger ● routing-table rib-only ● import-rib (BGP-VPN instance address family view) ● import-rib vpn-instance (BGP public network address family view) ● ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf ● ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 ● import-route (IS-IS) ● default-route-advertise (IS-IS) ● ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis ● preference (IS-IS) ● frr-policy route (IS-IS) ● filter-policy export (IS-IS) ● filter-policy import (IS-IS) ● import-route isis level-1 into level-2 (IS-IS) ● import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
{ isis-cost rip-cost ospf-cost } eq <i>cost</i>	Matches the IS-IS/RIP/OSPF routes with the specified <i>cost</i> .	<i>cost</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (OSPF) • default-route-advertise (OSPF) • preference (OSPF) • preference (OSPFv3) • local-mt filter-policy (OSPF) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • frr-policy route (OSPF) • frr-policy route (OSPFv3) • filter-policy import (OSPF) • filter-policy import (OSPFv3) • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • frr-policy route (IS-IS) • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • preference (RIP) • import-route (RIP) • default-route originate (RIP) • preference (RIPng) • import-route (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
{ isis-cost rip-cost ospf-cost } ge <i>cost</i>	Matches the IS-IS/RIP/OSPF routes with the cost greater than or equal to the specified <i>cost</i> .	<i>cost</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (OSPF) • default-route-advertise (OSPF) • preference (OSPF) • preference (OSPFv3) • local-mt filter-policy (OSPF) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • frr-policy route (OSPF) • frr-policy route (OSPFv3) • filter-policy import (OSPF) • filter-policy import (OSPFv3) • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • frr-policy route (IS-IS) • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • preference (RIP) • import-route (RIP) • default-route originate (RIP) • preference (RIPng) • import-route (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
{ isis-cost rip-cost ospf-cost } le cost	Matches the IS-IS/RIP/OSPF routes with the cost less than or equal to the specified <i>cost</i> .	<i>cost</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (OSPF) • default-route-advertise (OSPF) • preference (OSPF) • preference (OSPFv3) • local-mt filter-policy (OSPF) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • frr-policy route (OSPF) • frr-policy route (OSPFv3) • filter-policy import (OSPF) • filter-policy import (OSPFv3) • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • frr-policy route (IS-IS) • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • preference (RIP) • import-route (RIP) • default-route originate (RIP) • preference (RIPng) • import-route (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
rib-cost eq <i>cost</i>	Matches all routes with the specified <i>cost</i> in the IP routing table.	<i>cost</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import-route (OSPF) • default-route-advertise (OSPF) • import-route (IS-IS) • default-route-advertise (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis
rib-cost ge <i>cost</i>	Matches all routes with the cost greater than or equal to the specified <i>cost</i> in the IP routing table.	<i>cost</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import-route (OSPF) • default-route-advertise (OSPF) • import-route (IS-IS) • default-route-advertise (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
rib-cost le cost	Matches all routes with the cost less than or equal to the specified <i>cost</i> in the IP routing table.	<i>cost</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none">• import-route (BGP)• network (BGP)• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis• import-route (OSPF)• default-route-advertise (OSPF)• import-route (IS-IS)• default-route-advertise (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
preference eq <i>preference</i>	Matches the routes with the specified <i>preference</i> .	<i>preference</i> : indicates an integer ranging from 0 to 255.	<ul style="list-style-type: none"> • import-route (BGP) • route-filter import (BGP) • route-filter export (BGP) • peer route-filter import • peer route-filter export • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • ingress-lsp trigger • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPF) • import-route (OSPF) • import-route (OSPFv3) • preference (OSPF) • preference (OSPFv3) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • preference (RIP) • preference (RIPng) • import-route (RIP) • import-route (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• default-route originate (RIP)• ripng default-route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
preference le <i>preference</i>	Matches the routes whose priority is less than or equal to the specified <i>preference</i> .	<i>preference</i> : indicates an integer ranging from 0 to 255.	<ul style="list-style-type: none"> • import-route (BGP) • route-filter import (BGP) • route-filter export (BGP) • peer route-filter import • peer route-filter export • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • ingress-lsp trigger • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPF) • import-route (OSPF) • import-route (OSPFv3) • preference (OSPF) • preference (OSPFv3) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • preference (RIP) • preference (RIPng) • import-route (RIP) • import-route (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• default-route originate (RIP)• ripng default-route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
preference <i>preference</i>	Matches the routes whose priority is greater than or equal to the specified <i>preference</i> .	<i>preference</i> : indicates an integer ranging from 0 to 255.	<ul style="list-style-type: none"> • import-route (BGP) • route-filter import (BGP) • route-filter export (BGP) • peer route-filter import • peer route-filter export • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • ingress-lsp trigger • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPF) • import-route (OSPF) • import-route (OSPFv3) • preference (OSPF) • preference (OSPFv3) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • preference (RIP) • preference (RIPng) • import-route (RIP) • import-route (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• default-route originate (RIP)• ripng default-route

Condition Clauses That Are Based on the Source IP Address, Destination IP Address, or Next Hop IP Address of a Route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ip route-source in { ip-prefix-list-name ip-prefix-list }	Matches the address of the source that advertises the IPv4 route.	<p><i>ip-prefix-list-name</i>: indicates the name of an IPv4 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ip-prefix-list</i>: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths. For example, { 1.1.1.0 24, 2.2.2.2 32 }.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public address family view) • import-rib { public vpn-instance vpn-instance-name } protocol ospf • import-rib { public vpn-instance vpn-instance-name } protocol ospfv3 • import-rib { public vpn-instance vpn-instance-name } protocol isis • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPF) • filter-policy import (OSPFv3) • local-mt filter-policy (OSPF) • frr-policy route (OSPF) • frr-policy route (OSPFv3)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• import-route (OSPF)• import-route (OSPFv3)• preference (OSPF)• preference (OSPFv3)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• import-route (IS-IS)• default-route-advertise (IS-IS)• preference (IS-IS)• frr-policy route (IS-IS)• filter-policy export (IS-IS)• filter-policy import (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis• preference (RIP)• preference (RIPng)• import-route (RIP)• import-route (RIPng)• default-route originate (RIP)• import-route isis level-1 into level-2 (IS-IS)• import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ipv6 route-source in { ipv6-prefix-list-name ipv6-prefix-list }	Matches the address of the source that advertises the IPv6 route.	<p><i>ipv6-prefix-list-name</i>: indicates the name of an IPv6 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ipv6-prefix-list</i>: indicates the IPv6 prefix set in the format of {element A, element B...}, in which the elements are IPv6 addresses carrying mask lengths.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • import route-filter • export route-filter • qos-local-id • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • preference (OSPF) • preference (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPF) • filter-policy import (OSPFv3) • local-mt filter-policy (OSPF) • frr-policy route (OSPF) • frr-policy route (OSPFv3) • import-route (OSPF) • import-route (OSPFv3) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• import-route (IS-IS)• default-route-advertise (IS-IS)• preference (IS-IS)• frr-policy route (IS-IS)• filter-policy export (IS-IS)• filter-policy import (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis• preference (RIPng)• import-route (RIPng)• import-route isis level-1 into level-2 (IS-IS)• import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ip route-destination in { ip-prefix-list-name ip-prefix-list }	Matches the IPv4 routes carrying the specified destination IP address.	<p><i>ip-prefix-list-name</i>: indicates the name of an IPv4 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ip-prefix-list</i>: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • route-filter import (BGP) • route-filter export (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • preference (OSPF) • preference (OSPFv3)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• default-route-advertise (OSPF)• default-route-advertise (OSPFv3)• filter-policy import (OSPF)• filter-policy import (OSPFv3)• local-mt filter-policy (OSPF)• frr-policy route (OSPF)• frr-policy route (OSPFv3)• import-route (OSPF)• import-route (OSPFv3)• default-route-advertise (OSPF)• local-mt filter-policy (OSPF)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• import-route (IS-IS)• default-route-advertise (IS-IS)• preference (IS-IS)• frr-policy route (IS-IS)• filter-policy export (IS-IS)• filter-policy import (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis• preference (RIP)• preference (RIPng)• import-route (RIP)• import-route (RIPng)• default-route originate (RIP)• import-route isis level-1 into level-2 (IS-IS)• import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ipv6 route-destination in { ipv6-prefix-list-name ipv6-prefix-list }	Matches the IPv6 routes carrying the specified destination IPv6 address.	<p><i>ipv6-prefix-list-name</i>: indicates the name of an IPv6 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ipv6-prefix-list</i>: indicates the IPv6 prefix set in the format of {element A, element B...}, in which the elements are IPv6 addresses carrying mask lengths.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • route-filter import (BGP) • route-filter export (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import route-filter • export route-filter • qos-local-id • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • preference (OSPF) • preference (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• filter-policy import (OSPF)• filter-policy import (OSPFv3)• local-mt filter-policy (OSPF)• frr-policy route (OSPF)• frr-policy route (OSPFv3)• import-route (OSPF)• import-route (OSPFv3)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• import-route (IS-IS)• default-route-advertise (IS-IS)• preference (IS-IS)• frr-policy route (IS-IS)• filter-policy export (IS-IS)• filter-policy import (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis• preference (RIPng)• import-route (RIPng)• import-route isis level-1 into level-2 (IS-IS)• import-route isis level-2 into level-1 (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ip next-hop in { ip-prefix-list-name ip-prefix-list }	Matches the IPv4 routes carrying the specified next hop IP address.	<p><i>ip-prefix-list-name</i>: indicates the name of an IPv4 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ip-prefix-list</i>: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • preference (OSPF) • preference (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPF)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• filter-policy import (OSPFv3)• local-mt filter-policy (OSPF)• frr-policy route (OSPF)• frr-policy route (OSPFv3)• import-route (OSPFv3)• import-route (OSPF)• default-route-advertise (OSPF)• local-mt filter-policy (OSPF)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• filter-policy import (OSPF)• frr-policy route (OSPF)• import-route (IS-IS)• default-route-advertise (IS-IS)• preference (IS-IS)• frr-policy route (IS-IS)• filter-policy export (IS-IS)• filter-policy import (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis• import-route isis level-1 into level-2 (IS-IS)• import-route isis level-2 into level-1 (IS-IS)• preference (RIP)• preference (RIPng)• import-route (RIP)• import-route (RIPng)• default-route originate (RIP)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ipv6 next-hop in { ipv6-prefix-list-name ipv6-prefix-list }	Matches the IPv6 routes carrying the specified next hop IPv6 address.	<p><i>ipv6-prefix-list-name</i>: indicates the name of an IPv6 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ipv6-prefix-list</i>: indicates the IPv6 prefix set in the format of {element A, element B...}, in which the elements are IPv6 addresses carrying mask lengths.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import route-filter • export route-filter • qos-local-id • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • preference (OSPF) • preference (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • filter-policy import (OSPFv3) • local-mt filter-policy (OSPF) • frr-policy route (OSPFv3) • import-route (OSPFv3)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none">• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• import-route (IS-IS)• default-route-advertise (IS-IS)• preference (IS-IS)• frr-policy route (IS-IS)• filter-policy export (IS-IS)• filter-policy import (IS-IS)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis• preference (RIPng)• import-route (RIPng)• import-route isis level-1 into level-2 (IS-IS)• import-route isis level-2 into level-1 (IS-IS)

Condition Clauses That Are Based on the Tag of an OSPF, IS-IS or Static Route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
tag eq tag	Matches the routes with the specified <i>tag</i> .	<i>tag</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none">• import-route (BGP)• network (BGP)• peer route-filter import• peer route-filter export• preference (BGP)• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3• import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis• import route-filter• export route-filter• import-route (OSPF)• import-route (OSPFv3)• default-route-advertise (OSPF)• default-route-advertise (OSPFv3)• preference (OSPF)• preference (OSPFv3)• frr-policy route (OSPF)• frr-policy route (OSPFv3)• local-mt filter-policy (OSPF)• ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf• ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3• filter-policy import (OSPF)• filter-policy import (OSPFv3)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none"> • import-route (IS-IS) • default-route-advertise (IS-IS) • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • default-route originate (RIP) • import-route (RIP) • preference (RIP) • import-route (RIPng) • preference (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
tag ge tag	Matches the routes with the tag greater than or equal to the specified tag.	<i>tag</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import route-filter • export route-filter • import-route (OSPF) • import-route (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • preference (OSPF) • preference (OSPFv3) • frr-policy route (OSPF) • frr-policy route (OSPFv3) • local-mt filter-policy (OSPF) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • filter-policy import (OSPF) • filter-policy import (OSPFv3) • import-route (IS-IS) • default-route-advertise (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none"> • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • default-route originate (RIP) • import-route (RIP) • preference (RIP) • import-route (RIPng) • preference (RIPng)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
tag le tag	Matches the BGP routes with the tag less than or equal to the specified <i>tag</i> .	<i>tag</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import route-filter • export route-filter • import-route (OSPF) • import-route (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • preference (OSPF) • preference (OSPFv3) • frr-policy route (OSPF) • frr-policy route (OSPFv3) • local-mt filter-policy (OSPF) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • filter-policy import (OSPF) • filter-policy import (OSPFv3) • import-route (IS-IS) • default-route-advertise (IS-IS)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none"> • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • default-route originate (RIP) • import-route (RIP) • preference (RIP) • import-route (RIPng) • preference (RIPng)

Condition Clauses That Are Based on the Local_Pref Attribute of a BGP Route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
local-preference eq <i>preference</i>	Matches the BGP routes with the Local_Pref value being <i>preference</i> .	<i>preference</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export (This command takes effect only for IBGP peers.)
local-preference ge <i>preference</i>	Matches the BGP routes with the Local_Pref value greater than or equal to <i>preference</i> .	<i>preference</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export (This command takes effect only for IBGP peers.)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
local-preference le preference	Matches the BGP routes with the Local_Pref value less than or equal to <i>preference</i> .	<i>preference</i> : indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export (This command takes effect only for IBGP peers.)

Condition Clauses That Are Based on the AS_Path Attribute of a BGP Route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path in { as-path-list-name as-path-list }	Matches the BGP routes carrying the AS_Path in the specified AS_Path set.	<i>as-path-list-name</i> : indicates the name of an AS_Path set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:). <i>as-path-list</i> : indicates the AS_Path set in the format of {element A, element B...}, in which the elements are in the format of regular+AS_Path regular expression.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path is-none	Matches the BGP routes carrying an empty AS_Path (locally generated BGP routes).	-	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)
as-path origin <i>as-path</i> [whole-match]	Matches BGP routes with AS_Path whose rightmost AS numbers are the same as <i>as-path</i> .	<i>as-path</i> : The value is enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. Duplicate AS numbers are counted as one unless whole-match is configured.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path pass as-path [whole-match]	Matches BGP routes with AS_Path whose contiguous AS numbers match <i>as-path</i> .	<i>as-path</i> : The value is enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. Duplicate AS numbers are counted as one unless whole-match is configured.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)
as-path peer-is as-path [whole-match]	Matches BGP routes with AS_Path whose leftmost AS numbers are the same as <i>as-path</i> .	<i>as-path</i> : The value is enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. Duplicate AS numbers are counted as one unless whole-match is configured.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path length { eq ge le } as-path-number	<p>Matches the BGP routes carrying the AS_Path with the length equal to <i>as-path-number</i>, greater than or equal to <i>as-path-number</i>, or less than or equal to <i>as-path-number</i>.</p> <ul style="list-style-type: none"> • eq: equal to • ge: greater than or equal to • le: less than or equal to 	<p><i>as-path-number</i>: indicates the AS_Path length. The value is an integer ranging from 0 to 2047.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path unique-length { eq ge le } as-path-number	<p>Matches the BGP routes carrying the AS_Path with the length equal to <i>as-path-number</i>, greater than or equal to <i>as-path-number</i>, or less than or equal to <i>as-path-number</i> (duplicate AS numbers are counted as one).</p> <ul style="list-style-type: none"> • eq: equal to • ge: greater than or equal to • le: less than or equal to 	<p><i>as-path-number</i>: indicates the AS_Path length. The value is an integer ranging from 0 to 2047.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clauses That Are Based on the Community Attribute of a BGP Route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
community matches-any { community-list-name community-list }	Matches the BGP routes carrying at least one of the communities specified in the community set.	<i>community-list-name</i> : indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:). <i>community-list</i> : indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).	<ul style="list-style-type: none">● peer route-filter import● peer route-filter export● preference (BGP)● aggregate suppress-filter● aggregate origin-filter● dampening● import route-filter● export route-filter● qos-local-id● ingress-lsp trigger● routing-table rib-only● import-rib (BGP-VPN instance address family view)● import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
community matches-all { <i>community-list-name</i> <i>community-list</i> }	Matches the BGP routes carrying the communities listed in the specified community set or more communities.	<i>community-list-name</i> : indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:). <i>community-list</i> : indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
community matches-within { <i>community-list-name</i> <i>community-list</i> }	Matches the BGP routes carrying the communities listed in the specified community set or less communities.	<p><i>community-list-name</i>: indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>community-list</i>: indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).</p>	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
community is-none	Matches the BGP routes with an empty community.	-	<ul style="list-style-type: none">• peer route-filter import• peer route-filter export• preference (BGP)• aggregate suppress-filter• aggregate origin-filter• dampening• import route-filter• export route-filter• qos-local-id• ingress-lsp trigger• routing-table rib-only• import-rib (BGP-VPN instance address family view)• import-rib vpn-instance (BGP public network address family view)

Condition Clauses That Are Based on an Attribute of a VPN Route

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
rd in { rd-list-name rd-list }	Matches the VPN routes with RDs in the specified RD set.	<p><i>rd-list-name</i>: indicates the name of an RD set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rd-list</i>: indicates the RD set in the format of {element A, element B...}. For details about RD element values, see RD element values.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • nexthop recursive-lookup bit-error-detection
extcommunity rt matches-any { rt-list-name rt-list }	Matches the VPN routes carrying at least one of the RTs specified in the route target set.	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. For details about RT element values, see RT element values.</p>	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity rt matches-all { <i>rt-list-name</i> <i>rt-list</i> }	Matches the VPN routes carrying the RTs listed in the specified route target set or more RTs.	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. For details about RT element values, see RT element values.</p>	<ul style="list-style-type: none">● peer route-filter import● peer route-filter export● preference (BGP)● aggregate suppress-filter● aggregate origin-filter● dampening● import route-filter● export route-filter● qos-local-id● ingress-lsp trigger● routing-table rib-only● import-rib (BGP-VPN instance address family view)● import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity rt matches-within { rt-list-name rt-list }	Matches the VPN routes carrying the RTs listed in the specified route target set or less RTs.	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. For details about RT element values, see RT element values.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export ● preference (BGP) ● aggregate suppress-filter ● aggregate origin-filter ● dampening ● import route-filter ● export route-filter ● qos-local-id ● ingress-lsp trigger ● routing-table rib-only ● import-rib (BGP-VPN instance address family view) ● import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity rt is-none	Matches the VPN routes with an empty RT.	-	<ul style="list-style-type: none">• peer route-filter import• peer route-filter export• preference (BGP)• aggregate suppress-filter• aggregate origin-filter• dampening• import route-filter• export route-filter• qos-local-id• ingress-lsp trigger• routing-table rib-only• import-rib (BGP-VPN instance address family view)• import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommu nity soo matches- any { <i>soo- list-name</i> <i>soo-list</i> }	Matches the VPN routes carrying at least one of the site or origins (SoOs) specified in the SoO set.	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. For details about SoO element values, see SoO element values.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export ● preference (BGP) ● aggregate suppress-filter ● aggregate origin-filter ● dampening ● import route-filter ● export route-filter ● qos-local-id ● ingress-lsp trigger ● routing-table rib-only ● import-rib (BGP-VPN instance address family view) ● import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommu nity soo matches- all { <i>soo- list-name</i> <i>soo-list</i> }	Matches the VPN routes carrying the SoOs listed in the specified SoO set or more SoOs.	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. For details about SoO element values, see SoO element values.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export ● preference (BGP) ● aggregate suppress-filter ● aggregate origin-filter ● dampening ● import route-filter ● export route-filter ● qos-local-id ● ingress-lsp trigger ● routing-table rib-only ● import-rib (BGP-VPN instance address family view) ● import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommu nity soo matches- within { <i>soo-list- name</i> <i>soo-list</i> }	Matches the VPN routes carrying the SoOs listed in the specified SoO set or less SoOs.	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. For details about SoO element values, see SoO element values.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export ● preference (BGP) ● aggregate suppress-filter ● aggregate origin-filter ● dampening ● import route-filter ● export route-filter ● qos-local-id ● ingress-lsp trigger ● routing-table rib-only ● import-rib (BGP-VPN instance address family view) ● import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity soo is-none	Matches the VPN routes with an empty SoO.	-	<ul style="list-style-type: none">• peer route-filter import• peer route-filter export• preference (BGP)• aggregate suppress-filter• aggregate origin-filter• dampening• import route-filter• export route-filter• qos-local-id• ingress-lsp trigger• routing-table rib-only• import-rib (BGP-VPN instance address family view)• import-rib vpn-instance (BGP public network address family view)

Condition Clauses That Are Based on Any of Other Attributes

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
mpls-label exist	Matches the routes carrying MPLS labels.	-	<ul style="list-style-type: none">• import-route (BGP)• network (BGP)• peer route-filter import• peer route-filter export• preference (BGP)• nexthop recursive-lookup (BGP)• nexthop recursive-lookup bit-error-detection• import route-filter• export route-filter• qos-local-id• ingress-lsp trigger• routing-table rib-only• import-rib (BGP-VPN instance address family view)• import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
origin is <i>origin-type</i>	Matches the BGP routes carrying the specified <i>origin-type</i> .	The value of <i>origin-type</i> can be any of the following items: <ul style="list-style-type: none">• egp: indicates the routes learned through the EGP.• igp: indicates the routes that are added to the BGP routing table using the network (BGP) command.• incomplete: indicates the routes imported using the import-route (BGP) command.	<ul style="list-style-type: none">• import-route (BGP)• network (BGP)• peer route-filter import• peer route-filter export• aggregate suppress-filter• aggregate origin-filter• dampening• import route-filter• export route-filter• qos-local-id• ingress-lsp trigger• routing-table rib-only• import-rib (BGP-VPN instance address family view)• import-rib vpn-instance (BGP public network address family view)
protocol is <i>protocol-type</i>	Matches the routes of the specified routing protocol.	The value of <i>protocol-type</i> can be any of the following items: <ul style="list-style-type: none">• bgp• direct• isis• ospf• rip• static	<ul style="list-style-type: none">• network (BGP)• peer route-filter import• peer route-filter export• aggregate suppress-filter• aggregate origin-filter• nexthop recursive-lookup (BGP)
protocol in <i>protocol-type</i>	Matches the routes of the specified routing protocol.	The value of <i>protocol-type</i> can be any of the following items: <ul style="list-style-type: none">• bgp• direct• isis• ospf• rip• static	<ul style="list-style-type: none">• network (BGP)• peer route-filter import• peer route-filter export• aggregate suppress-filter• aggregate origin-filter• nexthop recursive-lookup (BGP)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
route-type is route-type	Matches the routes of the specified type.	<p>The value of <i>route-type</i> can be any of the following items:</p> <ul style="list-style-type: none"> • external-type1: indicates Type 1 external OSPF routes. • external-type2: indicates Type 2 external OSPF routes. • internal: indicates internal routes, including inter-area and intra-area OSPF routes. • nssa-external-type1: indicates Not-So-Stubby Area (NSSA) Type 1 external OSPF routes. • nssa-external-type2: indicates NSSA Type 2 external OSPF routes. • is-is-level-1: indicates Level-1 IS-IS routes. • is-is-level-2: indicates Level-2 IS-IS routes. 	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • preference (BGP) • aggregate suppress-filter • aggregate origin-filter • dampening • import route-filter • export route-filter • qos-local-id • ingress-lsp trigger • routing-table rib-only • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view) • import-route (OSPF) • import-route (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • preference (OSPF) • preference (OSPFv3) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • frr-policy route (OSPF) • frr-policy route (OSPFv3) • local-mt filter-policy (OSPF) • filter-policy import (OSPF)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			<ul style="list-style-type: none"> • filter-policy import (OSPFv3) • import-route (IS-IS) • default-route-advertise (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • preference (IS-IS) • frr-policy route (IS-IS) • filter-policy export (IS-IS) • filter-policy import (IS-IS)
path-type <i>is path-type</i>	Matches the BGP routes of the specified type.	The value of <i>path-type</i> can be either of the following items: <ul style="list-style-type: none"> • ibgp: indicates IBGP routes. • ebgp: indicates EBGP routes. 	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export
rplki <i>is rplki-type</i>	Matches the BGP routes with the specified origin AS validation result.	The value of <i>rplki-type</i> can be any of the following items: <ul style="list-style-type: none"> • invalid: indicates that the origin AS is incorrect. • not-found: indicates that the origin AS is not found. • valid: indicates that the origin AS is correct. 	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
orf-prefix in { ip-prefix-list-name ip-prefix-list }	Matches the BGP routes with the outbound route filtering (ORF) prefixes in the specified IPv4 prefix set.	<p><i>ip-prefix-list-name</i>: indicates the name of an IPv4 prefix set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>ip-prefix-list</i>: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths.</p>	None
interface in interface-list-name	Matches the BGP routes of which the outbound interfaces are in the specified outbound interface list.	<p><i>interface-list-name</i>: indicates the name of an outbound interface list. The value is a string of 1 to 64 case-sensitive characters, spaces not supported. The string can contain letters, digits, underscores (_), hyphens (-), and dots (.). It must start with a letter or digit. <i>interface-list-name</i> must have been set using the xpl interface-list section-name command.</p>	<ul style="list-style-type: none"> • import-route (BGP) • route-filter import (BGP) • route-filter export (BGP) • peer route-filter export • aggregate suppress-filter • aggregate origin-filter • dampening • nexthop recursive-lookup (BGP) • nexthop recursive-lookup bit-error-detection • import route-filter • export route-filter • ingress-lsp trigger • import-rib (BGP-VPN instance address family view) • import-rib vpn-instance (BGP public network address family view)

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
route-state is bgp-not-best	Matches non-optimal BGP routes.	bgp-not-best: indicates non-optimal BGP routes.	peer route-filter export

Action Clauses

Action clauses are used to set actions or route attributes for the routes that match condition clauses. This section describes different types of action clauses.

NOTE

The parameters in action clauses can be specific values or global variables referenced using \$+global variable name. The parameters in a route-filter with pre-defined variables can be those defined in this route-filter.

Global variables cannot be referenced by the route attribute sets in the format of {element A, element B...}.

Action clauses (excluding **approve**, **refuse**, **finish**, **call route-filter** *route-filter-name*, and **break**) must follow **apply**.

Action Clauses Used to Specify Actions in a Route-Filter

Action Clauses	Function	Parameters
approve	Further filters the routes that meet the matching rules of the current if condition branch against the next if condition branch; if the current if condition branch is the last one, these routes match the route-filter.	-
refuse	Denies routes that meet matching rules of a route-filter.	-
finish	Completes route filtering and indicates that the route matches the route-filter.	-

Action Clauses	Function	Parameters
call route-filter <i>route-filter-name</i>	References another route-filter.	The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. If parameters are included in the referenced route-filter, specify values for them in the format of (var1, var2, ...var8) behind the route-filter name. A maximum of eight parameters can be specified, and each value ranges from 1 to 200 characters.
break	Enables the device to exit from the current route-filter. If the current route-filter is referenced by a parent route-filter, the device keeps implementing remaining condition and action clauses of the parent route-filter.	-

Action Clauses Used to Set Common Protocol Attributes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply ip next-hop { <i>ipv4-address</i> blackhole peer-address }	Sets a next hop IPv4 address.	<p><i>ipv4-address</i>: indicates an IPv4 address in dotted decimal notation.</p> <p>blackhole: adds a black-hole flag to a route.</p> <p>peer-address: sets the next hop of a BGP route to the IPv4 address of a peer. The IP address is in dotted decimal notation.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● import-route (OSPF) ● default-route-advertise (OSPF) ● import-route (RIP) ● default-route originate (RIP) <p>The <i>ipv4-address</i> and peer-address parameters are invalid in the following commands:</p> <ul style="list-style-type: none"> ● import-route (BGP) ● network (BGP) <p>The blackhole parameter is invalid in following commands:</p> <ul style="list-style-type: none"> ● peer route-filter export <p>NOTE In the same route-filter, ip next-hop <i>ipv4-address</i> and ip next-hop peer-address commands cannot be both configured. Otherwise, the result may be unexpected.</p>
apply ipv6 next-hop { <i>ipv6-address</i> blackhole peer-address }	Sets a next hop IPv6 address.	<p><i>ipv6-address</i>: indicates an IPv6 address. The value is a 32-digit hexadecimal number, in the format of X:X:X:X:X:X:X:X.</p> <p>blackhole: adds a black-hole flag to a route.</p> <p>peer-address: sets the next hop of a BGP route to the IPv4 address of a peer.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● import-route (RIPng) <p>The <i>ipv6-address</i> and peer-address parameters are invalid in the following commands:</p> <ul style="list-style-type: none"> ● import-route (BGP) ● network (BGP) ● import route-filter <p>The blackhole parameter is invalid in following commands:</p> <ul style="list-style-type: none"> ● peer route-filter export <p>NOTE In the same route-filter, ipv6 next-hop <i>ipv6-address</i> and ipv6 next-hop peer-address commands cannot be both configured. Otherwise, the result may be unexpected.</p>

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply preference <i>preference</i>	<p>Sets a preference value for a routing protocol. When multiple routing protocols run on a router, routing information needs to be shared and selected among the routing protocols. Therefore, a default preference is specified for each routing protocol. When different routing protocols discover multiple routes to the same destination, the route discovered by the protocol with the highest priority is selected to forward IP packets. The smaller the</p> <p><i>preference</i>: indicates a route preference value. The value is an integer ranging from 1 to 255. The smaller the value, the higher the priority.</p>		<ul style="list-style-type: none"> • preference (BGP) • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • preference (OSPF) • ip import-rib <i>vpn-instance</i> <i>vpn-instance-name</i> protocol ospf • preference (OSPFv3) • preference (IS-IS) • ip import-rib <i>vpn-instance</i> <i>vpn-instance-name</i> protocol isis • preference (RIP) • preference (RIPng) • default-route originate (RIP)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
	preference value, the higher the priority.		
apply rib-cost [+ - inherit] rib-cost	Sets a cost for routes in the IP routing table.	+: increases the cost value. -: decreases the cost value. inherit-cost: inherits the costs of the routes. rib-cost: indicates a cost which is an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis import-route (OSPF) default-route-advertise (OSPF) default-route-advertise (OSPFv3) import-route (IS-IS) default-route-advertise (IS-IS) ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis
apply aigp { [+ -] cost inherit-cost }	Sets the AIGP value.	+: increases the AIGP value of routes. -: reduces the AIGP value of routes. <i>cost:</i> indicates the AIGP value. inherit-cost: inherits the costs of the routes as AIGP values.	<ul style="list-style-type: none"> import-route (BGP) network (BGP) <p>The inherit-cost parameter is invalid in following commands:</p> <ul style="list-style-type: none"> peer route-filter import peer route-filter export
apply mpls-label	Enables MPLS to allocate labels to routes.	-	<ul style="list-style-type: none"> peer route-filter import peer route-filter export

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply cost-type <i>cost-type</i>	Sets the cost type.	<p><i>cost-type</i> can be any of the following items:</p> <ul style="list-style-type: none"> • external: sets the cost type to IS-IS external. • internal: sets the cost type to IS-IS internal or sets the MED values of the BGP routes received from EBGP peers to the cost values of the IGP routes to which the BGP routes recurse. • type1: sets the cost type to OSPF external Type 1. • type2: sets the cost type to OSPF external Type 2. • internal-inc-ibgp: sets the MED values of the BGP routes received from IBGP or EBGP peers to the cost values 	<ul style="list-style-type: none"> • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • peer route-filter export • import-route (OSPF) • import-route (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • import-route (IS-IS) • default-route-advertise (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>of the IGP routes to which the BGP routes recurse.</p> <ul style="list-style-type: none"> ● med-plus-igp: sets the MED values of BGP routes by adding the original MED and the cost values of the IGP routes to which the BGP routes recurse. This parameter takes effect both on EBGP and IBGP peers. ● med-inherit-aigp: enables the MED of a BGP route to inherit the AIGP. 	
apply gateway-ip { origin-nexthop ip-address }	Sets a gateway IP address for a route.	<p>origin-nexthop: sets the gateway IPv4 address to the next-hop IPv4 address of the route.</p> <p><i>ip-address</i>: sets the gateway IPv4 address to a specified IPv4 address.</p>	<ul style="list-style-type: none"> ● peer route-filter import ● peer route-filter export

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply ipv6 gateway-ip { origin-nexthop ipv6-address }	Sets a gateway IPv6 address for a route.	origin-nexthop: sets the gateway IPv6 address to the next-hop IPv6 address of the route. ipv6-address: sets the gateway IPv6 address to a specified IPv6 address.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export

Action Clauses Used to Set Community Attributes for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply community { community-list-name community-list } overwrite	Overwrites the original community attribute with the specified community set.	<p>community-list-name: indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p>community-list: indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply community { community-list-name community-list } additive	Adds the specified community set to the original community attribute.	<p><i>community-list-name</i>: indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>community-list</i>: indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)
apply community in { community-list-name community-list } delete	Deletes the communities specified in the community set from the original community attribute.	<p><i>community-list-name</i>: indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>community-list</i>: indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply community not in { community-list-name community-list } delete	Deletes the community attributes that are not in the specified community set.	<p><i>community-list-name</i>: indicates the name of a community set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>community-list</i>: indicates the community set in the format of {element A, element B...}, in which the elements are in the format of aa:nn, a community number, or a known community (internet, no-export-subconfed, no-advertise, or no-export).</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)
apply community none	Deletes all community attributes from BGP routes.	-	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses Used to Set the AS_Path Attribute for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply as-path <i>as-path</i> overwrite	Sets the AS_Path attribute for BGP routes.	<i>as-path</i> : indicates the AS_Path, enclosed in single quotation marks, with every two neighboring AS numbers separated with a space.	<ul style="list-style-type: none"> ● import-route (BGP) ● network (BGP) ● peer route-filter import ● peer route-filter export ● aggregate attribute-filter ● peer default-route-advertise ● import route-filter ● export route-filter ● import-rib (BGP-VPN instance address family view) ● import-rib (BGP public network address family view)
apply as-path { as-number-plain as-number-dot } [<i>count</i>] additive	Adds an AS number to the AS_Path attribute of BGP routes.	as-number-plain: indicates an AS number which is an integer ranging from 1 to 4294967295. as-number-dot: indicates an AS number in the format of <i>x.y</i> , where <i>x</i> and <i>y</i> are integers ranging from 1 to 65535 and from 0 to 65535, respectively. <i>count</i> : indicates the number of times that an AS number is added to the AS_Path attribute of BGP routes.	<ul style="list-style-type: none"> ● import-route (BGP) ● network (BGP) ● peer route-filter import ● peer route-filter export ● aggregate attribute-filter ● peer default-route-advertise ● import route-filter ● export route-filter ● import-rib (BGP-VPN instance address family view) ● import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply as-path as-path substitute	Replaces the specified peer AS number in the AS_Path attribute with the local AS number.	<i>as-path</i> : indicates the AS_Path, enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. An AS_Path can include an AS number range enclosed in brackets, such as [12..23], which indicates any number from 12 to 23.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)
apply as-path as-path substitute whole-match	Replaces the specified peer AS number in the AS_Path attribute with the local AS number in the case of whole-match.	<i>as-path</i> : indicates the AS_Path, enclosed in single quotation marks, with every two neighboring AS numbers separated with a space. An AS_Path can include an AS number range enclosed in brackets, such as [12..23], which indicates any number from 12 to 23.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply as-path most-recent <i>most-recent-value</i>	Appends the most recent AS number in the existing AS_Path to the beginning of the AS_Path for the specified number of times.	<i>most-recent-value</i> : specifies the number of times the most recent AS number is appended to the beginning of the AS_Path. The value ranges from 1 to 32.	<ul style="list-style-type: none">• import-route (BGP)• network (BGP)• peer route-filter import• peer route-filter export• aggregate attribute-filter• peer default-route-advertise• import route-filter• export route-filter• import-rib (BGP-VPN instance address family view)• import-rib vpn-instance (BGP public network address family view)

Action Clauses Used to Set Attributes Related to BGP Route Selection

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply med { [+ -] med igr-cost }	Sets a value for the MED attribute of BGP routes.	+: increases the MED value. -: decreases the MED value. <i>med</i> indicates an integer ranging from 0 to 4294967295. When + or - is specified, <i>med</i> indicates the value to be increased or reduced by. igr-cost : enables a device to use the cost of each received IGP route as the MED of the BGP route to be advertised to BGP peers.	<ul style="list-style-type: none"> peer route-filter export import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis <p>The igr-cost parameter is invalid in following commands:</p> <ul style="list-style-type: none"> import-route (BGP) network (BGP) peer route-filter import aggregate attribute-filter peer default-route-advertise import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol direct import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static import route-filter export route-filter import-rib (BGP-VPN instance address family view) import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply local-preference [+ -] local-preference	Sets the Local_Pref attribute for BGP routes. The larger the value, the higher the priority.	<p><i>local-preference</i>: specifies the Local_Pref value of a BGP route. The value is an integer that ranges from 0 to 4294967295.</p> <p>+: increases the Local_Pref value.</p> <p>-: decreases the Local_Pref value.</p>	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export • aggregate attribute-filter • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)
apply preferred-value preferred-value	Sets the PrefVal attribute for BGP routes. During route selection, the route with the largest PrefVal is preferred.	<i>preferred-value</i> : indicates the PrefVal attribute. The value is an integer ranging from 0 to 65535.	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • aggregate attribute-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply origin <i>origin-type</i>	Sets the origin type for BGP routes.	<i>origin-type</i> can be any of the following items: <ul style="list-style-type: none">• egp: indicates that routes are learned through the EGP protocol.• igp: indicates the routes that are added to the BGP routing table using the network (BGP) command.• incomplete: indicates the routes imported using the import-route (BGP) command.	<ul style="list-style-type: none">• import-route (BGP)• network (BGP)• peer route-filter import• peer route-filter export• aggregate attribute-filter• peer default-route-advertise• import route-filter• export route-filter• import-rib (BGP-VPN instance address family view)• import-rib (BGP public network address family view)

Action Clauses Used to Set BGP-Specific Attributes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply dampening <i>half-life reuse suppress max-suppress</i>	Sets dampening parameters for BGP routes.	<p><i>half-life</i>: indicates the half life of reachable routes. The value is an integer ranging from 1 to 45, in minutes.</p> <p><i>reuse</i>: indicates a threshold for releasing routes from the dampening state. The threshold is an integer ranging from 1 to 20000. When the penalty value of a route falls below the threshold, the route is reused.</p> <p><i>suppress</i>: indicates the threshold to dampen routes. When the penalty value of a route exceeds the threshold, the route is dampened. The threshold is an integer ranging from 1 to 20000 and must be greater than <i>reuse</i>.</p> <p><i>max-suppress</i>: indicates the maximum penalty value. The value is an integer ranging from 1001 to 20000 and must be greater than <i>suppress</i>.</p>	<ul style="list-style-type: none"> • dampening
apply group <i>group-name</i> NOTE Only the NetEngine 8000 X supports QPPB.	Sets a QPPB configuration group.	<i>group-name</i> : indicates the name of a QPPB configuration group. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).	<ul style="list-style-type: none"> • peer route-filter import • import route-filter • qos-local-id • filter-policy import (OSPF) • local-mt filter-policy (OSPF)

Action Clauses Used to Set IGP-Specific Attributes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply ospf-cost [+ - inherit] ospf-cost	Sets a cost value for OSPF routes.	+: increases the cost value. -: decreases the cost value. inherit-cost: inherits the costs of the routes. <i>ospf-cost:</i> indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 default-route-advertise (OSPF) default-route-advertise (OSPFv3) import-route (OSPF) import-route (OSPFv3) ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 filter export (OSPF) filter import (OSPF) filter export (OSPFv3) filter import (OSPFv3)
apply isis-cost [+ - inherit] isis-cost	Sets a cost value for IS-IS routes.	+: increases the cost value. -: decreases the cost value. inherit-cost: inherits the costs of the routes. <i>isis-cost:</i> indicates an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> import-route (IS-IS) default-route-advertise (IS-IS)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply rip-cost [+ -] rip-cost	Sets a cost value for RIP routes.	+: increases the cost value. -: decreases the cost value. <i>rip-cost</i> : indicates the cost of RIP routes. The value is an integer ranging from 0 to 16.	<ul style="list-style-type: none"> • import-route (RIP) • import-route (RIPng) • default-route originate (RIP)
apply isis isis-level-type	Sets the level of the routes imported by IS-IS.	<i>isis-level-type</i> can be any of the following items: <ul style="list-style-type: none"> • level-1 • level-1-2 • level-2 	<ul style="list-style-type: none"> • import-route (IS-IS) • default-route-advertise (IS-IS)
apply ospf ospf-sub-type	Imports routes to a specified OSPF area.	<i>ospf-sub-type</i> can be either of the following items: <ul style="list-style-type: none"> • backbone: indicates the backbone area. • stub-area: indicates an NSSA. 	None

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply tag <i>tag</i>	Sets a tag for IGP routes.	<i>tag</i> : indicates the tag of IGP routes. The value is an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol static • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospf • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol ospfv3 • import-rib { public vpn-instance <i>vpn-instance-name</i> } protocol isis • import-route (OSPF) • import-route (OSPFv3) • default-route-advertise (OSPF) • default-route-advertise (OSPFv3) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol ospf • ipv6 import-rib vpn-instance <i>vpn-instance-name</i> protocol ospfv3 • import-route (IS-IS) • default-route-advertise (IS-IS) • ip import-rib vpn-instance <i>vpn-instance-name</i> protocol isis • import-route isis level-1 into level-2 (IS-IS) • import-route isis level-2 into level-1 (IS-IS) • import-route (RIP) • import-route (RIPng) • default-route originate (RIP)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply rip-tag <i>rip-tag</i>	Sets a tag for RIP routes.	<i>rip-tag</i> : indicates the tag of RIP routes. The value is an integer ranging from 0 to 65535.	<ul style="list-style-type: none">• import-route (OSPF)• import-route (OSPFv3)• default-route-advertise (OSPF)• default-route-advertise (OSPFv3)• import-route (RIP)• import-route (RIPng)• default-route originate (RIP)

Action Clauses Used to Set the RT Attribute for VPN Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity rt { rt-list-name rt-list } overwrite	Overwrites the original RT with the specified RT.	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>4294967295, and the user-defined number is an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity rt { rt-list-name rt-list } additive	Adds the specified RT to the original RT.	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is 	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • peer default-route-advertise • import route-filter • export route-filter • import-rib (BGP-VPN instance address family view) • import-rib (BGP public network address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity rt in { rt-list-name rt-list } delete	<p>Deletes the RTs that are contained in the specified route target set.</p>	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is 	None

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity rt not in { rt-list-name rt-list } delete	<p>Deletes the RTs that are not contained in the specified route target set.</p>	<p><i>rt-list-name</i>: indicates the name of a route target set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>rt-list</i>: indicates the route target set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is 	None

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	
apply extcomm unity rt none	Deletes all RTs from VPN routes.	-	None

Action Clauses Used to Set the SoO Attribute for VPN Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity soo { <i>soo-list-name</i> <i>soo-list</i> } overwrite	Overwrites the original SoO with the specified SoO.	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>4294967295, and the user-defined number is an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity soo { <i>soo-list-name</i> <i>soo-list</i> } additive	Adds the specified SoO to the original SoO.	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is 	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity soo in { soo-list-name soo-list } delete	Deletes the SoOs that are contained in the specified SoO set.	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is 	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export <p>NOTE The preceding commands can be run, but their configurations do not take effect.</p>

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none">• 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535.	

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity soo not in { <i>soo-list-name</i> <i>soo-list</i> } delete	<p>Deletes the SoOs that are not contained in the specified SoO set.</p>	<p><i>soo-list-name</i>: indicates the name of an SoO set. The value is a string of 1 to 200 case-sensitive characters, spaces and question marks not supported. The string can contain letters, digits, underscores (_), hyphens (-), dots (.), and colons (:). It must start with a letter, digit, or colon (:).</p> <p><i>soo-list</i>: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed in any of the following formats:</p> <ul style="list-style-type: none"> • 2-byte AS number:4-byte user-defined number, for example, 1:3. The AS number is an integer ranging from 0 to 65535, and the user-defined number is an integer ranging from 0 to 4294967295. • IPv4 address:2-byte user-defined number, for example, 192.168.122.15:1. The IPv4 address ranges from 0.0.0.0 to 255.255.255.255, and the user-defined number is an integer ranging from 0 to 65535. • Integral 4-byte AS number:2-byte user-defined number, for example, 0:3 or 65537:3. The integral 4-byte AS number ranges from 65536 to 4294967295, and the user-defined number is 	<ul style="list-style-type: none"> • import-route (BGP) • network (BGP) • peer route-filter import • peer route-filter export <p>NOTE The preceding commands can be run, but their configurations do not take effect.</p>

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		<p>an integer ranging from 0 to 65535.</p> <ul style="list-style-type: none"> 4-byte AS number in dotted notation:2-byte user-defined number, for example, 0.0:3 or 0.1:0. Generally, a 4-byte AS number in dotted notation is in the format of $x.y$, where x and y are both integers ranging from 0 to 65535. A user-defined number is an integer ranging from 0 to 65535. 	
apply extcomm unity soo none	Deletes all SoOs from VPN routes.	-	<ul style="list-style-type: none"> import-route (BGP) network (BGP) peer route-filter import peer route-filter export <p>NOTE The preceding commands can be run, but their configurations do not take effect.</p>

Action Clauses Used to Set Redirection Extended Community Attributes for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity redirect ip ip-extcomm unity-value	Sets the IP redirection extended community attribute for BGP routes.	<i>ip-extcommunity-value</i> : indicates the IP redirection extended community attribute in the format of <i>ip-address:nn</i> , in which <i>ip-address</i> is in dotted decimal notation and <i>nn</i> is 0 or 1.	<ul style="list-style-type: none"> peer route-filter import peer route-filter export import-route (BGP) network (BGP)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity redirect vpn-target vpnrt-extcomm unity-value	Sets the VPN redirection extended community attribute for BGP routes.	<i>vpnrt-extcommunity-value</i> : indicates the VPN redirection extended community attribute in the format of <i>as-number:nn</i> , in which <i>as-number</i> is an AS number with the value being an integer ranging from 0 to 65535 and <i>nn</i> is an integer ranging from 0 to 4294967295.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export

Setting a Color Extended Community Attribute for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity color extcomm unityvalue	Sets a color extended community attribute for BGP routes.	<i>extcommunityvalue</i> : The value is in the format of color flag:4-byte user-defined number. For example, the value can be set to 0:100. Currently, the color flag can only be 0, and the user-defined number ranges from 1 to 4294967295.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export • import route-filter • export route-filter

Setting a Link Bandwidth Extended Community Attribute for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
apply extcomm unity bandwidt h bandwidt h-extcomm unity-value overwrite	Sets a link bandwidth extended community attribute for BGP routes.	<i>bandwidth-extcommunity-value</i> : The value is in the format of 2-byte AS number:4-byte user-defined number, for example, 1:100. The 2-byte AS number ranges from 1 to 65535. You can also enter the special value as-trans , which indicates that the 2-byte AS number is 23456. The user-defined number indicates the link bandwidth and ranges from 1 to 4294967295, in kbit/s.	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export
apply extcomm unity bandwidt h none	Deletes the link bandwidth extended community attribute set for BGP routes.	-	<ul style="list-style-type: none"> • peer route-filter import • peer route-filter export
apply extcomm unity bandwidt h aggregat e limit bandwidt h-value	Specifies the maximum value of the bandwidth extended community attribute.	<i>bandwidth-value</i> : specifies the link bandwidth. The value is an integer that ranges from 1 to 4294967295, in kbit/s.	<ul style="list-style-type: none"> • peer route-filter export
apply extcomm unity bandwidt h aggregat e	Specifies the aggregate d bandwidth extended community attribute for BGP routes.	-	<ul style="list-style-type: none"> • peer route-filter export

1.1.13 Route Monitoring Group Configuration

1.1.13.1 Route Monitoring Group Description

1.1.13.1.1 Overview of Route Monitoring Groups

Definition

All monitored network-side routes of the same type can be added to a group called a route monitoring group. Each route monitoring group is identified by a unique name.

A route monitoring group monitors the status of its member routes, each of which has a down-weight. The down-weight indicates the link quality. The higher the value, the more important the route. The down-weight can be set based on parameters, such as the link bandwidth, rate, and cost.

- If a route in the route monitoring group goes Down, its down-weight is added to the down-weight sum of the route monitoring group.
- If a route in the route monitoring group goes Up again, its down-weight is subtracted from the down-weight sum of the route monitoring group.

Purpose

Service modules can be associated with the route monitoring group, with a threshold configured for each service module for triggering a primary/backup access-side link switchover. If the down-weight sum of the route monitoring group reaches the threshold of a service module, the routing management (RM) module notifies the service module to switch services from the primary link to the backup link. If the down-weight sum of the route monitoring group falls below the threshold, the RM module notifies the service module to switch services back.

Benefits

If a service module is associated with a route monitoring group in a dual-system backup scenario, services can be switched to the backup link if the primary link fails, therefore preventing traffic overload and forwarding failures.

1.1.13.1.2 Understanding Route Monitoring Groups

Route Monitoring Group Fundamentals

All monitored network-side routes of the same type can be added to a route monitoring group. A down-weight can be set for each route in the route monitoring group based on link attributes, such as the bandwidth, and a threshold can be set for each service module that is associated with the route monitoring group to trigger a primary/backup access-side link switchover.

- A route monitoring group monitors the status of its member routes. If a route in the group goes Down, its down-weight is added to the down-weight sum of the route monitoring group. If the down-weight sum of the route monitoring group reaches the threshold of a service module, the RM module

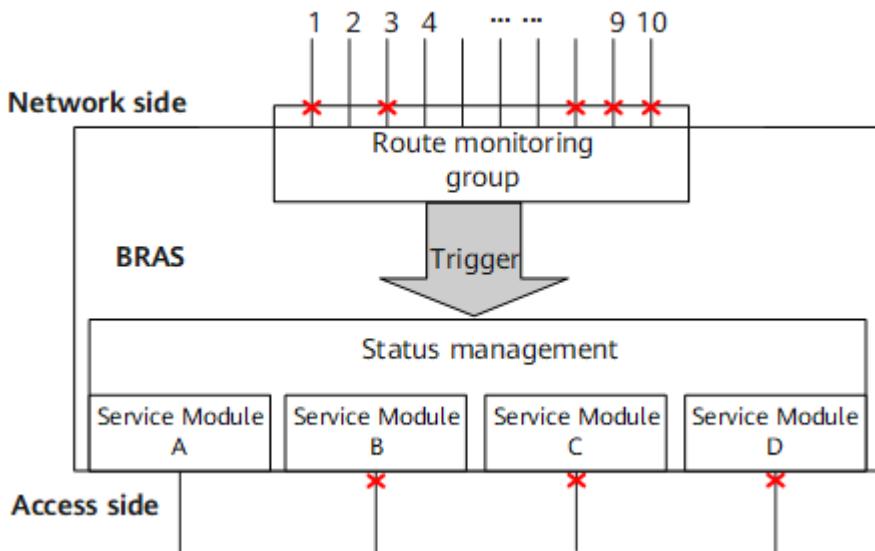
notifies the service module to switch services from the primary link to the backup link.

- If a route in the route monitoring group goes Up again, its down-weight is subtracted from the down-weight sum of the route monitoring group. If the down-weight sum of the route monitoring group falls below the threshold of a service module, the RM module notifies the service module to switch services back. You can specify the switchback delay time based on your actual network requirements.

As shown in **Figure 1-477**, the route monitoring group contains 10 routes, each having a down-weight of 10. The route monitoring group is associated with service modules A, B, C, and D whose thresholds for a primary/backup switchover are 80, 50, 30, and 20, respectively.

- If two routes in the route monitoring group go Down, the RM module notifies service module D to switch services from the primary link to the backup link. If one more route goes Down, the RM module notifies service module C to perform a primary/backup link switchover. If five routes go Down, the RM module notifies service module B to perform a primary/backup link switchover. If the number of routes that go Down reaches eight, the RM module notifies service module A to perform a primary/backup link switchover.
- If the number of routes that are Down in the route monitoring group falls below eight, the RM module notifies service module A to switch services back. If the number of routes that are Down in the route monitoring group falls below five, the RM module notifies service module B to switch services back. If the number of routes that are Down in the route monitoring group falls below three, the RM module notifies service module C to switch services back. If the number of routes that are Down in the route monitoring group falls below two, the RM module notifies service module D to switch services back.

Figure 1-477 Route monitoring group



1.1.13.1.3 Application Scenarios for Route Monitoring Groups

Applications of route monitoring groups

Service Overview

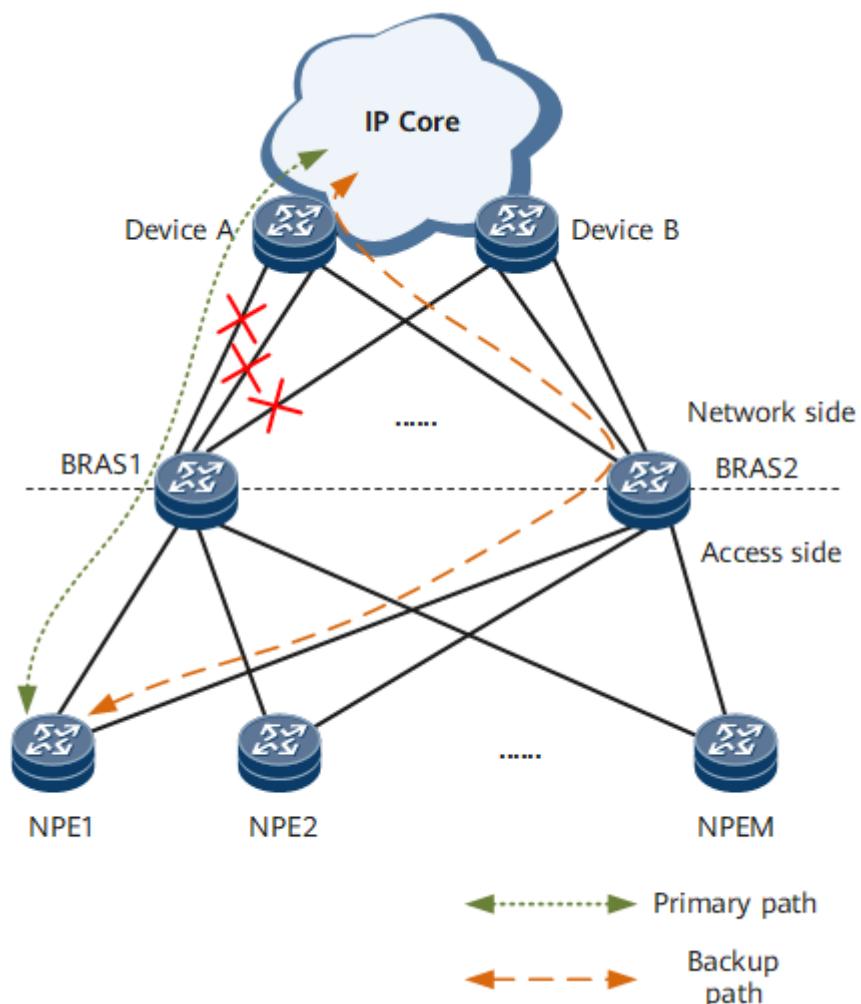
To improve network reliability, most carriers implement device-level redundancy by deploying two devices. The two devices back up each other or share traffic load. If one of the devices fails, the other device takes over services. Despite the benefit of enhanced network reliability, you must dual-home other devices to the two devices, which may introduce link reliability and load balancing issues.

Networking Description

As shown in [Figure 1-478](#), BRAS 2 backs up BRAS 1. NPEs on the user side are dual-homed to the two BRASs to load-balance traffic, and the two BRASs are connected to routers on the network side.

- If the link between BRAS 1 and Device A or between BRAS 1 and Device B fails, the link bandwidth between BRAS 1 and the IP core network decreases. The NPEs, however, cannot detect the link failure and keep sending packets to the IP core network through BRAS 1. As a result, the other link between BRAS 1 and the IP core network may be overloaded.
- If the links between BRAS 1 and Device A and between BRAS 1 and Device B both fail, only the links between BRAS 2 and the IP core network are available. The NPEs, however, cannot detect the link failure and keep sending packets to the IP core network through BRAS 1. As a result, the packets are discarded.

Figure 1-478 Route monitoring group



Feature Deployment

To address the packet drop issue, deploy a route monitoring group on each BRAS, and add network-side routes of the BRAS to the route monitoring group. If the down-weight sum of the route monitoring group reaches the threshold of a service module that is associated with the group, the RM module will notify the service module to trigger a primary/backup access-side link switchover. This mechanism prevents traffic overload and service interruptions.

1.1.13.1.4 Terminology for Route Monitoring Group

Terms

Term	Definition
Route monitoring group	A group that consists of monitored network-side routes of the same type.

1.1.13.2 Route Monitoring Group Configuration

1.1.13.2.1 Overview of Route Monitoring Groups

Definition

All monitored network-side routes of the same type can be added to a group called a route monitoring group. Each route monitoring group is identified by a unique name.

A route monitoring group monitors the status of its member routes, each of which has a down-weight. The down-weight indicates the link quality. The higher the value, the more important the route. The down-weight can be set based on parameters, such as the link bandwidth, rate, and cost.

- If a route in the route monitoring group goes Down, its down-weight is added to the down-weight sum of the route monitoring group.
- If a route in the route monitoring group goes Up again, its down-weight is subtracted from the down-weight sum of the route monitoring group.

Purpose

Service modules can be associated with the route monitoring group, with a threshold configured for each service module for triggering a primary/backup access-side link switchover. If the down-weight sum of the route monitoring group reaches the threshold of a service module, the routing management (RM) module notifies the service module to switch services from the primary link to the backup link. If the down-weight sum of the route monitoring group falls below the threshold, the RM module notifies the service module to switch services back.

Benefits

If a service module is associated with a route monitoring group in a dual-system backup scenario, services can be switched to the backup link if the primary link fails, therefore preventing traffic overload and forwarding failures.

1.1.13.2.2 Configuration Precautions for Route Monitoring Group

Feature Requirements

None

1.1.13.2.3 Configuring a Route Monitoring Group

You can add network-side routes to a route monitoring group and associate access-side service modules with it so that the service modules can perform primary/backup link switchovers upon route changes in the group in a dual-device hot backup scenario. This mechanism can prevent traffic congestion or loss.

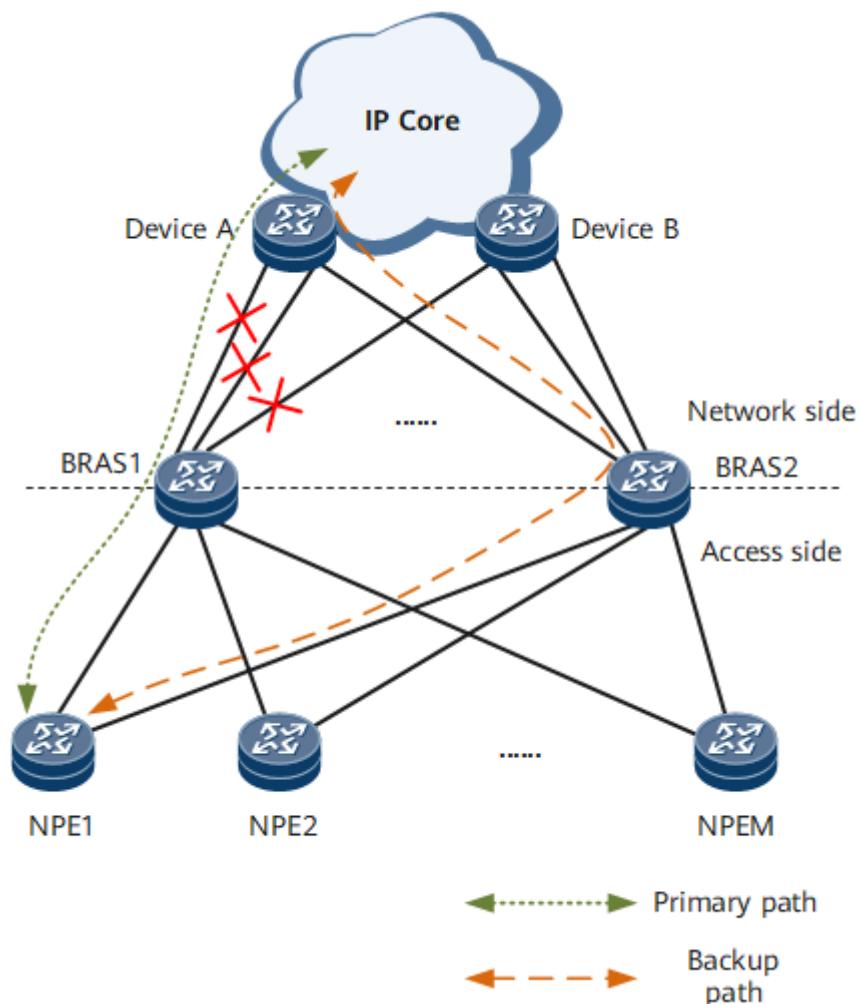
Context

To improve network reliability, most carriers implement device-level redundancy by deploying two devices. The two devices back up each other or share traffic load. If one of the devices fails, the other device takes over services. Despite the benefit of enhanced network reliability, you must dual-home other devices to the two devices, which may provoke link reliability and load balancing issues.

On the network shown in [Figure 1-479](#), BRAS 1 and BRAS 2 back up each other. NPEs on the user side are dual-homed to the two BRASs to load-balance traffic, and the network-side BRASs access the IP core network.

- If the link between BRAS 1 and router A or between BRAS 1 and router B fails, the link bandwidth between BRAS 1 and the IP core network decreases. The NPEs, however, cannot detect the link failure in time and keep sending packets to the IP core network through BRAS 1 (rather than BRAS 2). As a result, the other link between BRAS 1 and the IP core network may be overloaded.
- If the links between BRAS 1 and router A and between BRAS 1 and router B both fail, only the links between BRAS 2 and the IP core network are available. The NPEs, however, cannot detect the link failure in time and keep sending packets to the IP core network through BRAS 1 (rather than BRAS 2). As a result, the packets are discarded.

Figure 1-479 Route monitoring group



On the network shown in [Figure 1-479](#), deploy a route monitoring group on each BRAS, and add network-side routes of each BRAS to the route monitoring group. If a network-side link fails, the monitoring group monitors the status of network-side routes. When the status of a certain proportion of network-side routes on a BRAS changes, the RM module reports the change to the corresponding service module on the BRAS, triggering a primary/backup link switchover on the access side. Consequently, NPEs can access the network through another BRAS, preventing traffic loss and ensuring service continuity.

Procedure

- Configure a route monitoring group.
 - a. Run **system-view**
The system view is displayed.
 - b. Run **ip route-monitor-group *group-name***
A route monitoring group is created, and the route monitoring group view is displayed.
 - c. Run **track ip route [vpn-instance *vpn-instance-name*] destination { mask | mask-length } [down-weight *down-weight-value*]**

A route is added to the route monitoring group.

To add more routes to the route monitoring group, repeat this step. A maximum of 16 routes can be added to a route monitoring group.

d. (Optional) Run **trigger-up-delay** *delay-value*

A delay is configured for the RM module to instruct the service module associated with the route monitoring group to perform a switchback.

When a route in a route monitoring group becomes active again, the RM module needs to re-deliver the route to the forwarding table and create forwarding entries, which takes some time. If the RM module immediately instructs the service module to perform a link switchover, packet loss may occur. To prevent this problem, run the **trigger-up-delay** command to configure a delay for the RM module to instruct the service module to perform a link switchback so that the RM module instructs the service module to perform a link switchback after forwarding entries are created.

If the value of *delay-value* is set to 0, the RM module instructs the service module to perform a link switchback immediately when the down-weight sum of the route monitoring group falls below the switchback threshold.

e. Run **monitor enable**

The route monitoring group is enabled.

When a large number of routes are added to or deleted from a route monitoring group, the down-weight sum of the route monitoring group may change frequently, which in turn leads to service flapping of the service modules associated with the route monitoring group. To prevent such service flapping, run the **undo monitor enable** command to disable the route monitoring group so that it is dissociated from all service modules. After the configuration is complete, run the **monitor enable** command to re-associate the route monitoring group with the service modules.

f. (Optional) Run **quit**

Exit the route monitoring group view.

g. (Optional) Run **interface** *interface-type interface-number*

The access-side interface view is displayed.

h. (Optional) Run **track route-monitor-group** *groupName* [**trigger-down-weight** *downWeight*]

The interface is configured to track the route monitoring group.

An interface that tracks a route monitoring group is called a track interface. To allow multiple interfaces to track the same IPv4 route monitoring group, run this command on the interfaces one by one.

After an IPv4 route monitoring group is created and routes are added to the group, to use the IPv4 route monitoring group to trigger the status change of a user-side interface, run this command to associate the interface with the IPv4 route monitoring group. When the down-weight sum of the IPv4 route monitoring group reaches the set *downWeight* value, the status of the interface is set to Down, and services are switched

to the backup link. When the down-weight sum of the IPv4 route monitoring group falls below the configured *downWeight* value, the status of the interface is set to Up again, and services are switched back to the primary link. In this manner, the user-side interface can detect network-side route faults in time, ensuring user-side service reliability.

i. Run **commit**

The configuration is committed.

- Configure an IPv6 route monitoring group.

a. Run **system-view**

The system view is displayed.

b. Run **ipv6 route-monitor-group group-name**

An IPv6 route monitoring group is created, and the route monitoring group view is displayed.

c. Run **track ipv6 route [vpn-instance vpn-instance-name] destination mask-length [down-weight down-weight-value]**

An IPv6 route is added to the IPv6 route monitoring group.

To add more IPv6 routes to the route monitoring group, repeat this step. A maximum of 16 routes can be added to an IPv6 route monitoring group.

d. (Optional) Run **trigger-up-delay delay-value**

A delay is configured for the IPv6 RM module to instruct the service module associated with the IPv6 route monitoring group to perform a switchback.

When an IPv6 route in an IPv6 route monitoring group becomes active again, the IPv6 RM module needs to re-deliver the route to the forwarding table and create forwarding entries, which takes some time. If the IPv6 RM module immediately instructs the service module to perform a link switchover, packet loss may occur. To prevent this problem, run the **trigger-up-delay** command to configure a delay for the IPv6 RM module to instruct the service module to perform a link switchback so that the RM module instructs the service module to perform a link switchback after forwarding entries are created.

If the value of *delay-value* is set to 0, the IPv6 RM module instructs the service module to perform a link switchback immediately when the down-weight sum of the route monitoring group falls below the switchback threshold.

e. Run **monitor enable**

The IPv6 route monitoring group is enabled.

When a large number of IPv6 routes are added to or deleted from an IPv6 route monitoring group, the down-weight sum of the route monitoring group may change frequently, which in turn leads to service flapping of the service modules associated with the route monitoring group. To prevent such service flapping, run the **undo monitor enable** command to disable the IPv6 route monitoring group so that it is dissociated from all service modules. After the configuration is complete,

run the **monitor enable** command to re-associate the route monitoring group with the service modules.

f. (Optional) Run **quit**

Exit the route monitoring group view.

g. (Optional) Run **interface interface-type interface-number**

The access-side interface view is displayed.

h. (Optional) Run **track ipv6 route-monitor-group groupName [trigger-down-weight downWeight]**

The interface is configured to track the IPv6 route monitoring group.

An interface that tracks a route monitoring group is called a track interface. To allow multiple interfaces to track the same IPv6 route monitoring group, run this command on the interfaces one by one.

After an IPv6 route monitoring group is created and routes are added to the group, to use the IPv6 route monitoring group to trigger the status change of a user-side interface, run this command to associate the interface with the IPv6 route monitoring group. When the down-weight sum of the IPv6 route monitoring group reaches the set *downWeight* value, the status of the interface is set to Down, and services are switched to the backup link. When the down-weight sum of the IPv6 route monitoring group falls below the configured *downWeight* value, the status of the interface is set to Up again, and services are switched back to the primary link. In this manner, the user-side interface can detect network-side route faults in time, ensuring user-side service reliability.

i. Run **commit**

The configuration is committed.

----End

Example

After configuring the route monitoring group, check the configurations.

- Run the **display ip route-monitor-group [group-name]** command to check route monitoring group information.
- Run the **display ip route-monitor-group track-route [vpn-instance vpn-instance-name] dest-address { mask | mask-length }** command to check information about all route monitoring groups to which a route has been added.
- Run the **display ipv6 route-monitor-group [group-name]** command to check IPv6 route monitoring group information.
- Run the **display ipv6 route-monitor-group track-route [vpn-instance vpn-instance-name] destination mask-length** command to check information about all IPv6 route monitoring groups to which a route has been added.

Follow-up Procedure

Associate service modules with the status of a route monitoring group in a dual-device hot backup scenario. This prevents traffic overload and forwarding failures

when the primary link on the network side fails and thus improves user experience.

1.1.13.2.4 Configuration Examples for Route Monitoring Groups

Example for Configuring Association Between a Route Monitoring Group and a Dual-Device Hot Backup Module

In a dual-device hot backup scenario, if the primary link on the network side fails, the association between the service module and the route monitoring group prevents traffic overload and forwarding failures, improving user experience.

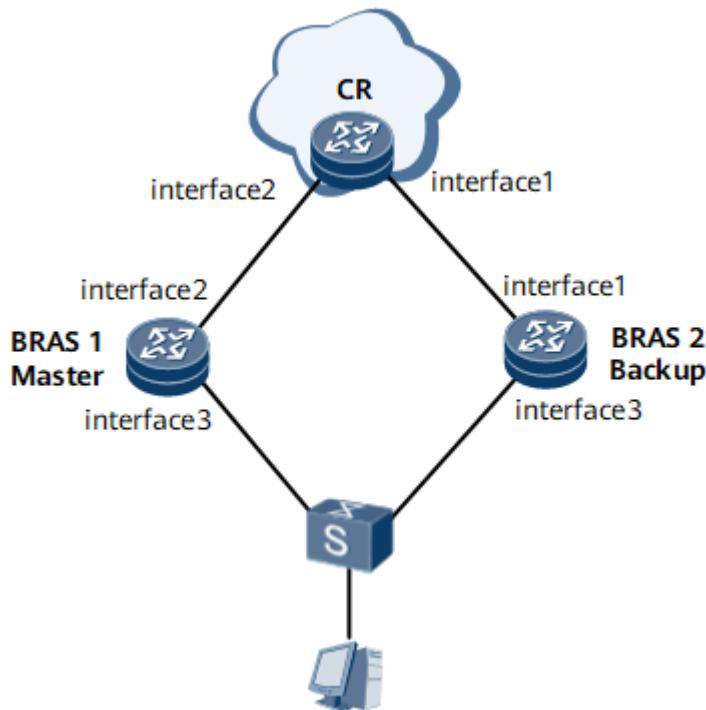
Networking Requirements

In the dual-device hot backup scenario shown in [Figure 1-480](#), you can add network-side routes to a route monitoring group and associate access-side service modules with it so that the service modules can perform primary/backup link switchovers upon route changes in the group. This mechanism prevents network congestion and packet loss.

Figure 1-480 Configuring association between a route monitoring group and a dual-device hot backup module

 NOTE

Interface 1, interface 2, and interface 3 in this example stand for GE 1/0/0, GE 2/0/0, and GE3/0/0.100 respectively.



Device Name	Interface	IP Address
BRAS1	LoopBack1	1.1.1.1/32
	GE 3/0/0.100	10.10.10.1/24
	GE 2/0/0	10.1.2.1/24
BRAS2	LoopBack1	2.2.2.2/32
	GE 1/0/0	10.1.4.1/24
	GE 3/0/0.100	10.10.10.10/24
CR	LoopBack1	3.3.3.3/32
	GE 1/0/0	10.1.4.2/24
	GE 2/0/0	10.1.2.2/24

Configuration Roadmap

The configuration roadmap is as follows:

1. Assign an IP address to each interface and configure routing protocols.
2. Configure a BFD session to rapidly detect faults in interfaces or links and trigger a master/backup VRRP switchover.
3. Configure a VRRP group on gigabitethernet 3/0/0.100, and configure the VRRP group to track the status of the BFD session and network-side interface.
4. Configure an RBS and an RBP.
5. Configure a route monitoring group on BRAS1 and associate the route monitoring group with the dual-device hot backup module.

Data Preparation

To complete the configuration, you need the backup ID, which works together with an RBS to identify an RBP to which users belong.

Procedure

- Step 1** Assign an IP address to each interface and configure device names and routing protocols.

For configuration details, see "Configuration Files" in this section.

- Step 2** Configure BFD.

```
[~BRAS1] bfd
[*BRAS1-bfd] quit
[*BRAS1] bfd atob bind peer-ip 10.10.10.10 source-ip 10.10.10.1
[*BRAS1-bfd-session-bfd] discriminator local 2
[*BRAS1-bfd-session-bfd] discriminator remote 3
[*BRAS1-bfd-session-bfd] commit
[~BRAS1-bfd-session-bfd] quit
```

The configuration of BRAS2 is similar to the configuration of BRAS1. For configuration details, see "Configuration Files" in this section.

- Step 3** Configure a VRRP group on an interface and configure the VRRP group to track the status of the BFD session and network-side interface.

```
[~BRAS1] interface gigabitethernet 3/0/0.100
[*BRAS1-Gigabitethernet3/0/0.100] vrrp vrid 3 virtual-ip 10.10.10.2
[*BRAS1-Gigabitethernet3/0/0.100] admin-vrrp vrid 3
[*BRAS1-Gigabitethernet3/0/0.100] vrrp vrid 3 priority 120
[*BRAS1-Gigabitethernet3/0/0.100] vrrp vrid 3 preempt-mode timer delay 1200
[*BRAS1-Gigabitethernet3/0/0.100] vrrp vrid 3 track bfd-session 2 peer
[*BRAS1-Gigabitethernet3/0/0.100] vrrp recover-delay 20
[*BRAS1-Gigabitethernet3/0/0.100] commit
[~BRAS1-Gigabitethernet3/0/0.100] quit
```

The configuration of BRAS2 is similar to the configuration of BRAS1. For configuration details, see "Configuration Files" in this section.

- Step 4** Configure an RBS and an RBP.

Configure an IP address for the RBS.

```
[~BRAS1] interface loopback1
[*BRAS1-loopback1] ip address 1.1.1.1 32
[*BRAS1-loopback1] commit
[~BRAS1-loopback1] quit
```

Configure a remote backup service.

```
[~BRAS1] remote-backup-service rbsv8
[*BRAS1-rm-backup-srv-rbsv8] peer 2.2.2.2 source 1.1.1.1 port 55333
[*BRAS1-rm-backup-srv-rbsv8] commit
[~BRAS1-rm-backup-srv-rbsv8] quit
```

Configure an RBP.

```
[~BRAS1] remote-backup-profile rbpv8
[*BRAS1-rm-backup-prf-rbpv8] service-type bras
[*BRAS1-rm-backup-prf-rbpv8] backup-id 1024 remote-backup-service rbsv8
[*BRAS1-rm-backup-prf-rbpv8] peer-backup hot
[*BRAS1-rm-backup-prf-rbpv8] vrrp-id 3 interface gigabitethernet 3/0/0.100
[*BRAS1-rm-backup-prf-rbpv8] commit
[~BRAS1-rm-backup-prf-rbpv8] quit
```

The configuration of BRAS2 is similar to the configuration of BRAS1. For configuration details, see "Configuration Files" in this section.

- Step 5** Configure a route monitoring group on BRAS1 and associate the route monitoring group with the dual-device hot backup module.

```
[~BRAS1] ip route-monitor-group lp
[*BRAS1-route-monitor-group-lp] track ip route 10.1.2.0 24 down-weight 20
[*BRAS1-route-monitor-group-lp] monitor enable
[*BRAS1-route-monitor-group-lp] commit
[~BRAS1-route-monitor-group-lp] quit
[~BRAS1] remote-backup-service rbsv8
[*BRAS1-rm-backup-srv-rbsv8] track route-monitor-group lp switchover failure-ratio 20
[*BRAS1-rm-backup-srv-rbsv8] commit
[~BRAS1-rm-backup-srv-rbsv8] quit
```

- Step 6** Verify the configuration.

Run the **display remote-backup-service** command on BRAS1. The command output shows that the route monitoring group is associated with the dual-device hot backup module.

```
[~BRAS1] display remote-backup-service rbsv8
2017-12-23 11:41:50.663
-----
Service-Index : 1
Service-Name  : rbsv8
```

```
TCP-State      : Connected
Peer-ip       : 2.2.2.2
Source-ip     : 1.1.1.1
TCP-Port      : 55333
Track-BFD     : -
SSL-Policy-Name : --
SSL-State     : --
Last up time   : 2017-12-21 15:33:06
Last down time  : 2017-12-21 16:31:10
Last down reason : TCP closed for echo time out
Uplink state   : 2 (1:DOWN 2:UP)
Track-route-monitor-group lp switchover percent 20
    TotalWeight : 0
    DownWeight  : 0
Domain-map-list : --
```

Run the **display ip route-monitor-group** command on BRAS1 to view information about the route monitoring group.

```
[~BRAS1] display ip route-monitor-group
Route monitor group number : 1
Route monitor group      Total weight      Down weight      State
lp                      10                  10                Enabled
```

----End

Configuration Files

- BRAS1 configuration file

```
#
sysname BRAS1
#
bfd
#
bfd atob bind peer-ip 10.10.10.10 source-ip 10.10.10.1
discriminator local 2
discriminator remote 3
#
ip route-monitor-group lp
track ip route 10.1.2.0 24 down-weight 20
monitor enable
#
remote-backup-service rbsv8
peer 2.2.2.2 source 1.1.1.1 port 55333
track route-monitor-group lp switchover failure-ratio 20
#
remote-backup-profile rbpv8
service-type bras
backup-id 1024 remote-backup-service rbsv8
peer-backup hot
vrrp-id 3 interface GigabitEthernet3/0/0.100
#
interface GigabitEthernet2/0/0
ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet3/0/0.100
vlan-type dot1q 600
ip address 10.10.10.1 255.255.255.0
vrrp vrid 3 virtual-ip 10.10.10.2
admin-vrrp vrid 3
vrrp vrid 3 priority 120
vrrp vrid 3 preempt-mode timer delay 1200
vrrp vrid 3 track bfd-session 2 peer
vrrp recover-delay 20
#
interface LoopBack1
ip address 1.1.1.1 255.255.255.255
```

```
#  
ospf 4  
import-route direct  
area 0.0.0  
network 10.1.2.0 0.0.0.255  
network 1.1.1.0 0.0.0.255  
#  
return
```

- BRAS2 configuration file

```
#  
sysname BRAS2  
#  
bfd  
#  
bfd atob bind peer-ip 10.10.10.1 source-ip 10.10.10.10  
discriminator local 3  
discriminator remote 2  
#  
remote-backup-service rbsv8  
peer 1.1.1.1 source 2.2.2.2 port 55333  
#  
remote-backup-profile rbpv8  
service-type bras  
backup-id 1024 remote-backup-service rbsv8  
peer-backup hot  
vrrp-id 3 interface GigabitEthernet3/0/0.100  
#  
interface GigabitEthernet1/0/0  
ip address 10.1.4.1 255.255.255.0  
#  
interface GigabitEthernet3/0/0.100  
vlan-type dot1q 600  
ip address 10.10.10.10 255.255.255.0  
vrrp vrid 3 virtual-ip 10.10.10.2  
admin-vrrp vrid 3  
vrrp vrid 3 track bfd-session 3 peer  
vrrp recover-delay 20  
#  
interface LoopBack1  
ip address 2.2.2.2 255.255.255.255  
#  
ospf 4  
import-route direct  
area 0.0.0  
network 10.1.4.0 0.0.0.255  
network 2.2.2.0 0.0.0.255  
#  
return
```

- CR configuration file

```
#  
sysname CR  
#  
interface GigabitEthernet1/0/0  
ip address 10.1.4.2 255.255.255.0  
#  
interface GigabitEthernet2/0/0  
ip address 10.1.2.2 255.255.255.0  
#  
interface LoopBack1  
ip address 3.3.3.3 255.255.255.255  
#  
ospf 4  
import-route direct  
area 0.0.0  
network 10.1.4.0 0.0.0.255  
network 10.1.2.0 0.0.0.255  
network 3.3.3.0 0.0.0.255
```

```
#  
return
```