HUAWEI NE40E-M2 series
V800R022C00SPC600

Configuration Guide

Issue	01
Date	2022-10-31

HUAWEI TECHNOLOGIES CO., LTD.

Copyright © Huawei Technologies Co., Ltd. 2022. 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 About This Document.....1
1.1.2 Basic IP Routing Configuration.....5
1.1.2.1 Overview of Basic IP Routing.....5
1.1.2.2 Configuration Precautions for Basic IP Routing.....6
1.1.2.3 Configuring the Router ID.....6
1.1.2.4 Configuring IPv4 Multi-Topology.....7
1.1.2.5 Configuring IPv6 Multi-Topology.....8
1.1.2.6 Configuring IPv4 FRR.....8
1.1.2.6.1 Enabling IPv4 FRR.....9
1.1.2.6.2 (Optional) Enabling IPv4 FRR Poison Reverse.....10
1.1.2.6.3 Verifying the Configuration.....12
1.1.2.7 Configuring IPv6 FRR.....12
1.1.2.7.1 Enabling IPv6 FRR.....13
1.1.2.7.2 (Optional) Enabling IPv6 FRR Poison Reverse.....13
1.1.2.7.3 Verifying the Configuration.....15
1.1.2.8 Configuring Route Recursion to the Default Route.....15
1.1.2.9 Configuring the Device to Advertise Host Routes of Directly Connected Interfaces.....16
1.1.2.10 Configuring Association Between Direct Routes and a VRRP Group.....17
1.1.2.10.1 Associating Direct Routes with a VRRP Group.....17
1.1.2.10.2 Configuring a Dynamic Routing Protocol to Import Direct Routes.....18
1.1.2.10.3 Verifying the Configuration of Association Between Direct Routes and a VRRP Group.....20
1.1.2.11 Configuring an IPv4 Direct Route to Respond to L3VE Interface Status Changes After a Delay.....20
1.1.2.12 Configuring an IPv6 Direct Route to Respond to L3VE Interface Status Changes After a Delay.....23
1.1.2.13 Configuring the Association Between the IPv4 Direct Route and PW Status.....25
1.1.2.14 Configuring the Association Between the IPv6 Direct Route and PW Status.....27
1.1.2.15 Configuring AGGs to Load Balance Downstream Traffic.....30
1.1.2.16 Configuring the Advertisement of Public Network IPv4 ARP Vlink Direct Routes.....32
1.1.2.17 Configuring the Advertisement of IPv6 NDP Vlink Direct Routes on the Public Network.....34
1.1.2.18 Configuring the Device to Filter ARP or ND Entries When Advertising VPN Vlink Direct Routes.....36
1.1.2.19 Configuring Route or Tunnel Recursion Suppression in Case of Flapping.....38
1.1.2.20 Maintaining IP Routes.....39

1.1.2.20.1 Displaying the Routing Table ... 39
1.1.2.20.2 Clearing Routing Table Statistics ... 41
1.1.2.20.3 Configuring a Limit on the Number of IPv4 Public Route Prefixes ... 41
1.1.2.20.4 Configuring a Limit on the Number of IPv6 Public Route Prefixes ... 42
1.1.2.20.5 Configuring Thresholds for the Number of IPv4 Route Prefixes ... 43
1.1.2.20.6 Configuring Thresholds for the Number of IPv6 Route Prefixes ... 44
1.1.2.21 Configuration Examples for IP Routing Basis ... 45
1.1.2.21.1 Example for Configuring Public Network IPv4 FRR ... 45
1.1.2.21.2 Example for Configuring Public Network IPv6 FRR ... 49
1.1.2.21.3 Example for Associating Direct Routes with a VRRP Group ... 54
1.1.2.21.4 Example for Importing IPv4 ARP Vlink Direct Routes to BGP ... 59
1.1.2.21.5 Example for Importing IPv6 NDP Vlink Direct Routes to BGP4+ ... 66
1.1.3 IPv4 Static Route Configuration ... 75
1.1.3.1 Overview of IPv4 Static Routes ... 75
1.1.3.2 Configuration Precautions for IPv4 static route ... 76
1.1.3.3 Configuring IPv4 Static Routes ... 78
1.1.3.3.1 Creating IPv4 Static Routes ... 78
1.1.3.3.2 (Optional) Setting the Default Priority for IPv4 Static Routes ... 81
1.1.3.3.3 (Optional) Configuring Recursion Depth-based Static Route Selection ... 81
1.1.3.3.4 (Optional) Configuring a Device to Iterate Static Routes to ARP Vlink Routes ... 82
1.1.3.3.5 (Optional) Enabling a Device to Recurse Static Routes to SRv6 Routes ... 82
1.1.3.3.6 (Optional) Preventing an IPv4 Static Route from Being Selected If the BFD Session Associated with It Is in the AdminDown State ... 83
1.1.3.3.7 Verifying the IPv4 Static Route Configuration ... 84
1.1.3.4 Configuring IPv4 Floating Static Routes ... 84
1.1.3.5 Configuring Association between LDP and Static Routes ... 85
1.1.3.6 Configuring IPv4 Static Route Detection ... 88
1.1.3.6.1 Configuring Dynamic BFD for IPv4 Static Routes ... 89
1.1.3.6.2 Configuring Static BFD for IPv4 Static Routes ... 90
1.1.3.6.3 Configuring NQA for IPv4 Static Routes ... 92
1.1.3.6.4 Associating EFM with IPv4 Static Routes ... 94
1.1.3.7 Configuring FRR for IPv4 Static Routes ... 95
1.1.3.8 Configuration Examples for IPv4 Static Routes ... 97
1.1.3.8.1 Example for Configuring IPv4 Static Routes ... 97
1.1.3.8.2 Example for Configuring IPv4 Floating Static Routes ... 100
1.1.3.8.3 Example for Configuring Dynamic BFD for IPv4 Static Routes ... 104
1.1.3.8.4 Example for Configuring Static BFD for IPv4 Static Routes ... 107
1.1.3.8.5 Example for Configuring NQA for IPv4 Static Routes ... 110
1.1.3.8.6 Example for Associating EFM with IPv4 Static Routes ... 118
1.1.3.8.7 Example for Configuring FRR for IPv4 Static Routes on the Public Network ... 121
1.1.3.8.8 Example for Configuring Association between LDP and Static Routes ... 126
1.1.4 IPv6 Static Route Configuration ... 132
1.1.4.1 Overview of IPv6 Static Routes ... 132

1.1.4.2 Configuration Precautions for IPv6 Static Route 133
1.1.4.3 Configuring IPv6 Static Routes 133
1.1.4.3.1 Creating IPv6 Static Routes 133
1.1.4.3.2 (Optional) Setting the Default Priority for IPv6 Static Routes 134
1.1.4.3.3 (Optional) Enabling a Device to Recurse IPv6 Static Routes to ARP Vlink Direct Routes 135
1.1.4.3.4 (Optional) Enabling a Device to Recurse IPv6 Static Routes to SRv6 Routes 135
1.1.4.3.5 (Optional) Preventing an IPv6 Static Route from Being Selected If the BFD Session Associated with It Is in the AdminDown State 136
1.1.4.3.6 Verifying the IPv6 Static Route Configuration 137
1.1.4.4 Configuring IPv6 Floating Static Routes 137
1.1.4.5 Configuring IPv6 Static Route Detection 138
1.1.4.5.1 Configuring Dynamic BFD for IPv6 Static Routes 139
1.1.4.5.2 Configuring Static BFD for IPv6 Static Routes 141
1.1.4.5.3 Configuring NQA for IPv6 Static Routes 142
1.1.4.6 Configuring FRR for IPv6 Static Routes 145
1.1.4.7 Configuration Examples for IPv6 Static Routes 146
1.1.4.7.1 Example for Configuring IPv6 Static Routes 146
1.1.4.7.2 Example for Configuring IPv6 Floating Static Routes 150
1.1.4.7.3 Example for Configuring Dynamic BFD for IPv6 Static Routes 154
1.1.4.7.4 Example for Configuring Static BFD for IPv6 Static Routes 158
1.1.4.7.5 Example for Configuring NQA for IPv6 Static Routes 161
1.1.4.7.6 Example for Configuring FRR for IPv6 Static Routes on the Public Network 168
1.1.5 OSPF Configuration 175
1.1.5.1 Overview of OSPF 176
1.1.5.2 Configuration Precautions for OSPF 178
1.1.5.3 Configuring Basic OSPF Functions 181
1.1.5.3.1 Enabling OSPF 182
1.1.5.3.2 Configuring an Interface to Fill in a DD Packet with the Interface MTU 184
1.1.5.3.3 (Optional) Creating OSPF Virtual Links 185
1.1.5.3.4 (Optional) Configuring the Router to Comply with Route Selection Rules Defined in a Standard Protocol 186
1.1.5.3.5 (Optional) Setting the OSPF Priority 187
1.1.5.3.6 (Optional) Restricting the Flooding of OSPF Update LSAs 188
1.1.5.3.7 (Optional) Configuring the Maximum Number of Packet Retransmission Attempts 189
1.1.5.3.8 (Optional) Setting the Interval at Which LSAs Are Retransmitted Between Adjacent Routers 190
1.1.5.3.9 Verifying the Configuration 191
1.1.5.4 Configuring OSPF on the NBMA or P2MP Network 192
1.1.5.4.1 Configuring Network Types for OSPF Interfaces 193
1.1.5.4.2 Configuring NBMA Network Attributes 194
1.1.5.4.3 Configuring P2MP Network Attributes 195
1.1.5.4.4 Verifying the Configuration 197
1.1.5.5 Adjusting OSPF Route Selection 198
1.1.5.5.1 Setting the Interface Cost 198

1.1.5.5.2 Configuring Equal-Cost Routes.... 200
1.1.5.5.3 Setting the convergence priority for OSPF routes.... 201
1.1.5.5.4 Configuring a Stub Router.... 202
1.1.5.5.5 Suppressing an Interface from Receiving and Sending OSPF Packets.... 203
1.1.5.5.6 Configuring an OSPF Interface to Automatically Adjust the Link Cost.... 204
1.1.5.5.7 Configuring a Fallback Cost for an Eth-Trunk Interface.... 205
1.1.5.5.8 Verifying the Configuration.... 205
1.1.5.6 Controlling OSPF Routing Information.... 206
1.1.5.6.1 Configuring OSPF to Import External Routes.... 206
1.1.5.6.2 Configuring OSPF to Import a Default Route.... 209
1.1.5.6.3 Configuring Route Summarization.... 210
1.1.5.6.4 Configuring OSPF to Filter LSAs in an Area.... 211
1.1.5.6.5 Configuring OSPF to Filter LSAs to Be Sent.... 214
1.1.5.6.6 Configuring OSPF to Filter Received Routes.... 216
1.1.5.6.7 Configuring OSPF to Filter the Routes to Be Advertised.... 218
1.1.5.6.8 (Optional) Configuring OSPF to Discard Specified LSAs.... 219
1.1.5.6.9 Configuring the Maximum Number of External Routes Supported by the OSPF LSDB.... 221
1.1.5.6.10 Verifying the Configuration of OSPF Routing Information Control.... 221
1.1.5.7 Adjusting the OSPF Network Convergence Speed.... 221
1.1.5.7.1 Setting the Interval at Which Hello Packets Are Sent.... 222
1.1.5.7.2 Setting the Delay for Transmitting LSAs on an Interface.... 222
1.1.5.7.3 Setting the Dead Time of an OSPF Neighbor.... 223
1.1.5.7.4 Configuring OSPF Sham Hello.... 224
1.1.5.7.5 Configuring Smart-discover.... 224
1.1.5.7.6 Setting the Interval at Which LSAs Are Updated.... 225
1.1.5.7.7 Setting the Interval at Which LSAs Are Received.... 226
1.1.5.7.8 Setting the Interval for the SPF Calculation.... 227
1.1.5.7.9 Configuring the Function to Suppress the Advertisement of Interface Addresses.... 228
1.1.5.7.10 Configuring the Route Calculation Delay Function to Suppress Frequent LSA Flapping.... 229
1.1.5.7.11 Disabling Active/Standby Board Switching Triggered by Abnormal OSPF Aging.... 230
1.1.5.7.12 Disabling OSPF LSA Aging Management.... 230
1.1.5.7.13 Setting a Period During Which OSPF Keeps the Maximum Cost in Local LSAs.... 231
1.1.5.7.14 Configuring Secure Synchronization.... 231
1.1.5.7.15 Verifying the Configuration of the OSPF Network Convergence Speed.... 232
1.1.5.8 Configuring OSPF Delay Advertisement.... 232
1.1.5.9 Configuring OSPF Neighbor Relationship Flapping Suppression.... 233
1.1.5.10 Disabling OSPF Interface Flapping Suppression.... 236
1.1.5.11 Configuring Routing Loop Detection for Routes Imported into OSPF.... 236
1.1.5.12 Configuring OSPF Flush LSA Source Tracing.... 237
1.1.5.13 Configuring an OSPF Hostname.... 238
1.1.5.14 Configuring an OSPF Stub Area.... 239
1.1.5.15 Configuring an NSSA.... 241

1.1.5.16 Configuring OSPF Local MT ... 244
1.1.5.17 Configuring an OSPF Sham Link ... 247
1.1.5.18 Configuring BFD for OSPF ... 250
1.1.5.18.1 Configuring BFD Globally ... 250
1.1.5.18.2 Configuring BFD for an OSPF Process ... 251
1.1.5.18.3 (Optional) Configuring BFD for a Specified Interface ... 253
1.1.5.18.4 Verifying the Configuration of BFD for OSPF ... 255
1.1.5.19 Configuring OSPF IP FRR ... 255
1.1.5.19.1 Enabling OSPF IP FRR ... 257
1.1.5.19.2 (Optional) Binding IP FRR and BFD ... 260
1.1.5.19.3 (Optional) Disabling OSPF IP FRR on an Interface ... 261
1.1.5.19.4 Verifying the Configuration of OSPF IP FRR ... 261
1.1.5.20 Configuring OSPF GR Helper ... 262
1.1.5.21 Improving OSPF Network Security ... 264
1.1.5.21.1 Configuring Area Authentication ... 265
1.1.5.21.2 Configuring Interface Authentication ... 266
1.1.5.21.3 Verifying the Configuration of OSPF Network Security ... 267
1.1.5.22 Configuring the Network Management Function of OSPF ... 268
1.1.5.23 Configuring Whitelist Session-CAR for OSPF ... 268
1.1.5.24 Configuring Micro-Segment CAR for OSPF ... 269
1.1.5.25 Configuring the Threshold Alarm Function for the Number of OSPF Neighbors ... 270
1.1.5.26 Maintaining OSPF ... 271
1.1.5.26.1 Clearing OSPF Statistics ... 271
1.1.5.26.2 Resetting OSPF ... 271
1.1.5.26.3 Disabling OSPF Memory Overload Control ... 272
1.1.5.26.4 Disabling OSPF CPU Overload Control ... 273
1.1.5.27 Configuring OSPF Multi-Area Adjacency ... 273
1.1.5.27.1 Enabling OSPF on a Multi-Area Adjacency Interface ... 275
1.1.5.27.2 Configuring a Cost for a Multi-Area Adjacency Interface ... 276
1.1.5.27.3 Configuring an Authentication Mode for a Multi-Area Adjacency Interface ... 276
1.1.5.27.4 Configuring Fast Convergence for a Multi-Area Adjacency Interface ... 277
1.1.5.27.5 Configuring Neighbor Relationship Flapping Suppression on an OSPF Multi-Area Adjacency Interface ... 280
1.1.5.27.6 Disabling an OSPF Multi-Area Adjacency Interface from Creating BFD Sessions ... 282
1.1.5.27.7 Disabling OSPF IP FRR on an OSPF Multi-Area Adjacency Interface ... 283
1.1.5.27.8 Configuring a Fallback Cost for an Eth-Trunk Multi-Area Adjacency Interface ... 283
1.1.5.27.9 Maintaining OSPF on a Multi-Area Adjacency Interface ... 284
1.1.5.27.10 Verifying the Configuration of OSPF Multi-Area Adjacency ... 284
1.1.5.28 Configuration Examples for OSPF ... 285
1.1.5.28.1 Example for Configuring Basic OSPF Functions ... 285
1.1.5.28.2 Example for Configuring OSPF Virtual Links ... 292
1.1.5.28.3 Example for Configuring an OSPF Stub Area ... 296
1.1.5.28.4 Example for Configuring an OSPF NSSA ... 302

1.1.5.28.5 Example for Configuring OSPF DR Election ... 306
1.1.5.28.6 Example for Configuring OSPF Load Balancing ... 312
1.1.5.28.7 Example for Configuring OSPF Fast Convergence ... 317
1.1.5.28.8 Example for Configuring OSPF Local MT ... 321
1.1.5.28.9 Example for Configuring OSPF IP FRR ... 329
1.1.5.28.10 Example for Configuring BFD for OSPF ... 334
1.1.5.28.11 Example for Configuring OSPF-BGP Synchronization ... 340
1.1.5.28.12 Example for Configuring Routing Loop Detection for Routes Imported from BGP to OSPF ... 349
1.1.5.28.13 Example for Configuring OSPF Multi-Area Adjacency ... 356
1.1.5.28.14 Example for Configuring an OSPF Sham Link ... 361
1.1.6 OSPFv3 Configuration ... 372
1.1.6.1 OSPFv3 Overview ... 372
1.1.6.2 Configuration Precautions for OSPFv3 ... 373
1.1.6.3 Configuring Basic OSPFv3 Functions ... 374
1.1.6.3.1 Enabling OSPFv3 ... 375
1.1.6.3.2 Enabling OSPFv3 on an Interface ... 376
1.1.6.3.3 Creating an OSPFv3 Area ... 377
1.1.6.3.4 (Optional) Configuring the Router to Comply with Route Selection Rules Defined in a Standard Protocol ... 377
1.1.6.3.5 (Optional) Configuring the Maximum Number of Packet Retransmission Attempts ... 378
1.1.6.3.6 (Optional) Disabling an Interface from Receiving and Sending OSPFv3 Packets ... 379
1.1.6.3.7 Verifying the Basic OSPFv3 Configuration ... 379
1.1.6.4 Configuring OSPFv3 Attributes on Different Types of Networks ... 380
1.1.6.4.1 Configuring the Network Type for an Interface ... 381
1.1.6.4.2 (Optional) Setting the DR Priority for the OSPFv3 Broadcast or NBMA Interface ... 382
1.1.6.4.3 (Optional) Setting the Polling Interval at Which Hello Packets Are Sent on an NBMA Network ... 383
1.1.6.4.4 (Optional) Disabling MTU Check on DD Packets ... 384
1.1.6.4.5 Verifying the Configuration of OSPFv3 Attributes in Different Types of Networks ... 384
1.1.6.5 Adjusting OSPFv3 Route Selection ... 384
1.1.6.5.1 Setting the Link Cost on an OSPFv3 Interface ... 385
1.1.6.5.2 Setting the Maximum Number of Equal-Cost Routes ... 386
1.1.6.5.3 Setting the convergence priority for OSPFv3 routes ... 387
1.1.6.5.4 Configuring an OSPFv3 Interface to Automatically Adjust the Link Cost ... 387
1.1.6.5.5 Verifying the Configuration of OSPFv3 Route Selection ... 388
1.1.6.6 Controlling OSPFv3 Routing Information ... 389
1.1.6.6.1 Configuring OSPFv3 to Import External Routes ... 389
1.1.6.6.2 Configuring OSPFv3 to Advertise Default Routes to OSPFv3 Areas ... 390
1.1.6.6.3 Configuring OSPFv3 to Filter Received Routes ... 391
1.1.6.6.4 Configuring OSPFv3 to Filter the Routes to Be Advertised ... 393
1.1.6.6.5 Configuring OSPFv3 to Filter LSAs in an Area ... 395
1.1.6.6.6 (Optional) Configuring OSPFv3 to Discard Specified LSAs ... 398
1.1.6.6.7 Configuring OSPFv3 Route Summarization ... 399

1.1.6.6.8 Verifying the Configuration of OSPFv3 Routing Information Control.....401
1.1.6.7 Disabling DN Bit Setting in OSPFv3 LSAs.....401
1.1.6.8 Configuring OSPFv3 Neighbor Relationship Flapping Suppression.....403
1.1.6.9 Configuring Routing Loop Detection for Routes Imported to OSPFv3.....405
1.1.6.10 Disabling OSPFv3 Interface Flapping Suppression.....406
1.1.6.11 Configuring OSPFv3 Flush Source Tracing.....406
1.1.6.12 Configuring an OSPFv3 Hostname.....408
1.1.6.13 Configuring an OSPFv3 Stub Area.....409
1.1.6.14 Configuring an OSPFv3 NSSA.....411
1.1.6.15 Configuring OSPFv3 IP FRR.....413
1.1.6.15.1 Enabling OSPFv3 IP FRR.....414
1.1.6.15.2 (Optional) Binding IP FRR and BFD.....416
1.1.6.15.3 (Optional) Blocking FRR on an OSPFv3 Interface.....417
1.1.6.15.4 Verifying the Configuration of OSPFv3 IP FRR.....417
1.1.6.16 Configuring BFD for OSPFv3.....417
1.1.6.16.1 Configuring BFD Globally.....418
1.1.6.16.2 Configuring BFD for OSPFv3.....418
1.1.6.16.3 (Optional) Configuring BFD for a Specified Interface.....420
1.1.6.16.4 Verifying the Configuration of BFD for OPSFv3.....422
1.1.6.17 Configuring OSPFv3 Authentication.....422
1.1.6.17.1 Configuring OSPFv3 IPsec.....423
1.1.6.17.2 Configuring OSPFv3 Authentication Trailer.....426
1.1.6.17.3 Configuring OSPFv3 Process Authentication.....428
1.1.6.17.4 Configuring OSPFv3 Area Authentication.....429
1.1.6.17.5 Configuring OSPFv3 Interface Authentication.....430
1.1.6.17.6 Verifying the Configuration of OSPFv3 Authentication.....431
1.1.6.18 Configuring OSPFv3 Fast Convergence.....431
1.1.6.18.1 Setting the Interval at Which Hello Packets Are Sent.....431
1.1.6.18.2 Setting the Dead Time of the Neighbor Relationship.....432
1.1.6.18.3 Configuring OSPFv3 Sham Hello.....433
1.1.6.18.4 Setting the Interval at Which LSAs Are Updated.....433
1.1.6.18.5 Setting the Interval at Which LSAs Are Received.....434
1.1.6.18.6 Disabling OSPFv3 LSA Aging Management.....435
1.1.6.18.7 Verifying the Configuration of OSPFv3 Fast Convergence.....436
1.1.6.19 Configuring the OSPFv3 GR Helper.....436
1.1.6.20 Configuring OSPFv3 to Report Network Topology Information to BGP-LS.....437
1.1.6.21 Improving OSPFv3 Network Stability.....438
1.1.6.21.1 Setting the Priority of OSPFv3.....438
1.1.6.21.2 Setting the Delay in Transmitting LSAs on the Interface.....439
1.1.6.21.3 Setting the Interval at Which LSAs Are Retransmitted Between Adjacent Routers.....439
1.1.6.21.4 Configuring a Route Calculation Delay to Suppress Frequent LSA Flapping.....440
1.1.6.21.5 Configuring the SPF Timer.....441

1.1.6.21.6 Configuring a Stub Router....442
1.1.6.21.7 Verifying the Configuration of OSPFv3 Network Stability....443
1.1.6.21.8 Configuring the Network Management Function of OSPFv3....443
1.1.6.22 Whitelist Session-CAR for OSPFv3....444
1.1.6.23 Configuring Micro-Isolation CAR for OSPFv3....445
1.1.6.24 Configuring the Threshold Alarm Function for the Number of OSPFv3 Neighbors....446
1.1.6.25 Maintaining OSPFv3....446
1.1.6.25.1 Clearing OSPFv3 Statistics....446
1.1.6.25.2 Resetting OSPFv3....447
1.1.6.25.3 Disabling OSPFv3 Memory Overload Control....447
1.1.6.25.4 Disabling OSPFv3 CPU Overload Control....448
1.1.6.26 Configuration Examples for OSPFv3....448
1.1.6.26.1 Example for Configuring Basic OSPFv3 Functions....449
1.1.6.26.2 Example for Configuring OSPFv3 DR Election....453
1.1.6.26.3 Example for Configuring OSPFv3 IP FRR....458
1.1.6.26.4 Example for Configuring BFD for OSPFv3....463
1.1.6.26.5 Example for Configuring IPSec for OSPFv3....468
1.1.6.26.6 Example for Configuring OSPFv3 Route Summarization on an ASBR....473
1.1.6.26.7 Example for Configuring OSPFv3 Route Summarization on an ABR....478
1.1.6.26.8 Example for Configuring an OSPFv3 NSSA....485
1.1.6.26.9 Example for Configuring Routing Loop Detection for OSPFv3 Multi-Process Mutual Route Import....491
1.1.7 RIP Configuration....499
1.1.7.1 Overview of RIP....499
1.1.7.2 Configuration Precautions for RIP....500
1.1.7.3 Configuring Basic RIP Functions....501
1.1.7.3.1 Creating a RIP process....501
1.1.7.3.2 Enabling RIP on Specified Network Segments....501
1.1.7.3.3 (Optional) Configuring the RIP Version Number....502
1.1.7.3.4 (Optional) Configuring a RIP Preference....503
1.1.7.3.5 (Optional) Disabling Check on the Source Address of RIP Packets on a P2P Network....504
1.1.7.3.6 (Optional) Enabling Check on Zero Fields of RIP-1 Packets....504
1.1.7.3.7 Configuring an NBMA Network....505
1.1.7.3.8 Verifying the Basic RIP Configuration....506
1.1.7.4 Preventing Routing Loops....506
1.1.7.4.1 Configuring Split Horizon....507
1.1.7.4.2 Configuring Poison Reverse....507
1.1.7.4.3 Configuring Suppression Timers....508
1.1.7.4.4 Verifying the Configuration of Routing Loop Prevention....509
1.1.7.5 Adjusting RIP Route Selection....509
1.1.7.5.1 Disabling RIP-2 Classful Summarization....510
1.1.7.5.2 Configuring the Additional Metric on an Interface....511
1.1.7.5.3 Setting the Maximum Number of Equal-Cost Routes....513

1.1.7.5.4 Verifying the Configuration of RIP Route Selection....514
1.1.7.6 Controlling RIP Routing Information....514
1.1.7.6.1 Configuring RIP to Import External Routes....514
1.1.7.6.2 Configuring RIP to Advertise Default Routes....515
1.1.7.6.3 Configuring RIP to Filter Received Routes....516
1.1.7.6.4 Configuring RIP to Filter the Routes to Be Advertised....517
1.1.7.6.5 Disabling RIP from Receiving Host Routes....518
1.1.7.6.6 Disabling the Check on RIP Packets with Cost 0....519
1.1.7.6.7 Disabling an Interface from Sending Packets....520
1.1.7.6.8 Disabling an Interface from Receiving Packets....520
1.1.7.6.9 Verifying the Configuration of RIP Routing Information Control....521
1.1.7.7 Configuring RIP Fast Convergence....522
1.1.7.7.1 Configuring RIP-2 Route Summarization....522
1.1.7.7.2 Configuring RIP Timers....523
1.1.7.7.3 Configuring RIP Triggered Update....524
1.1.7.7.4 Setting the Interval at Which Packets Are Sent and the Maximum Number of the Sent Packets....524
1.1.7.7.5 Setting the Maximum Length of RIP packets....525
1.1.7.7.6 Setting the Maximum Number of RIP Routes....525
1.1.7.7.7 Verifying the Configuration of RIP Fast Convergence....526
1.1.7.8 Configuring Dynamic BFD for RIP....526
1.1.7.9 Configuring Static BFD for RIP....529
1.1.7.10 Configuring OPS to Control Packet Sending and Receiving on RIP Interfaces....530
1.1.7.11 Improving the RIP Network Security....532
1.1.7.11.1 Configuring an Authentication Mode for RIP-2 Packets....533
1.1.7.11.2 Configuring the Check on the Source Address in RIP Packets on the Broadcast Network....540
1.1.7.11.3 Configuring GTSM for RIP....540
1.1.7.11.4 Verifying the Configuration of RIP Network Security....541
1.1.7.12 Configuring RIP Network Management....542
1.1.7.13 Maintaining RIP....542
1.1.7.13.1 Clearing RIP....543
1.1.7.13.2 Debugging RIP....543
1.1.7.14 Configuration Examples for RIP....544
1.1.7.14.1 Example for Configuring Basic RIP Functions....544
1.1.7.14.2 Example for Preventing Routing Loops....548
1.1.7.14.3 Example for Configuring RIP to Import Routes....552
1.1.7.14.4 Example for Configuring Dynamic BFD for RIP....557
1.1.7.14.5 Example for Configuring Static BFD for RIP....562
1.1.8 RIPng Configuration....567
1.1.8.1 Overview of RIPng....567
1.1.8.2 Configuration Precautions for RIPng....568
1.1.8.3 Configuring Basic RIPng Functions....568
1.1.8.3.1 Creating a RIPng process....568

1.1.8.3.2 Enabling RIPng on an Interface....569
1.1.8.3.3 (Optional) Configuring the RIPng Priority....570
1.1.8.3.4 (Optional) Enabling Check on Zero Fields of RIPng Packets....570
1.1.8.3.5 Verifying the Basic RIPng Configuration....571
1.1.8.4 Controlling the Sending and Receiving of RIPng Packets....571
1.1.8.4.1 Disabling Receiving of RIPng Packets on an Interface....571
1.1.8.4.2 Disabling Sending of RIPng Packets on an Interface....572
1.1.8.4.3 Setting the Interval for Sending Update Packets and the Maximum Number of Packets Sent Each Time....572
1.1.8.4.4 Verifying the Configuration of Controlling the Sending and Receiving of RIPng Packets....573
1.1.8.5 Preventing Routing Loops....573
1.1.8.5.1 Configuring Split Horizon....574
1.1.8.5.2 Configuring Poison Reverse....575
1.1.8.5.3 Configuring Suppression Timers....575
1.1.8.5.4 Verifying the Configuration of Routing Loop Prevention....576
1.1.8.6 Adjusting RIPng Route Selection....576
1.1.8.6.1 Configuring the Additional Metric on an Interface....577
1.1.8.6.2 Setting the Maximum Number of Equal-Cost Routes....578
1.1.8.6.3 Verifying the Configuration of RIPng Route Selection....579
1.1.8.7 Controlling RIPng Routing Information....579
1.1.8.7.1 Configuring RIPng to Import External Routes....580
1.1.8.7.2 Configuring RIPng to Filter the Received Routes....580
1.1.8.7.3 Configuring RIPng to Filter the Routes to be Sent....582
1.1.8.7.4 Configuring RIPng to Advertise the Default Routes....583
1.1.8.7.5 Configuring RIPng Route Summarization....584
1.1.8.7.6 Verifying the Configuration of RIPng Routing Information Control....585
1.1.8.8 Configuring RIPng Timers....585
1.1.8.9 Configuring IPsec Authentication for RIPng....586
1.1.8.9.1 Configuring IPsec Authentication for a RIPng Process....586
1.1.8.9.2 Configuring IPsec Authentication on a RIPng Interface....587
1.1.8.9.3 Verifying the Configuration of IPsec Authentication for RIPng....588
1.1.8.10 Maintaining RIPng....588
1.1.8.10.1 Clearing RIPng....588
1.1.8.10.2 Debugging RIPng....588
1.1.8.11 RIPng Configuration Examples....589
1.1.8.11.1 Example for Configuring Basic RIPng Functions....589
1.1.8.11.2 Example for Configuring Split Horizon....594
1.1.9 IS-IS Configuration....598
1.1.9.1 Overview of IS-IS....598
1.1.9.2 Configuration Precautions for IS-IS....598
1.1.9.3 Configuring Basic IPv4 IS-IS Functions....601
1.1.9.3.1 Creating an IPv4 IS-IS Process....601
1.1.9.3.2 Configuring an IPv4 IS-IS Interface....605

1.1.9.3.3 (Optional) Configuring Costs for IPv4 IS-IS Interfaces ... 606
1.1.9.3.4 (Optional) Configuring IPv4 IS-IS Attributes on Networks of Different Types ... 610
1.1.9.3.5 (Optional) Configuring IS-IS to Adjust the Flooding Rate (IPv4) ... 613
1.1.9.3.6 (Optional) Enabling LSP Fragment Extension on an IS-IS Device (IPv4) ... 613
1.1.9.3.7 (Optional) Enabling the Type-132 TLV in LSPs to Carry the IPv4 Address of Only One IS-IS Interface ... 614
1.1.9.3.8 Verifying the Configuration ... 614
1.1.9.4 Configuring IPv4 IS-IS Route Selection ... 615
1.1.9.4.1 Configuring IPv4 IS-IS Route Leaking ... 616
1.1.9.4.2 Configuring Rules for Selecting Equal-Cost IPv4 IS-IS Routes ... 619
1.1.9.4.3 Filtering IPv4 IS-IS Routes ... 621
1.1.9.4.4 Configuring an Overload Bit for an IPv4 IS-IS Device ... 622
1.1.9.4.5 Configuring IS-IS to Generate IPv4 Default Routes ... 623
1.1.9.4.6 Configuring an IPv4 IS-IS Interface to Automatically Adjust the Link Cost ... 624
1.1.9.4.7 Configuring IPv4 IS-IS Route Recursion to IPv6 Next Hops ... 625
1.1.9.4.8 Verifying the IPv4 IS-IS Route Selection Configuration ... 626
1.1.9.5 Configuring IPv4 IS-IS to Interact with Other Routing Protocols ... 626
1.1.9.5.1 Configuring a Preference Value for IPv4 IS-IS ... 627
1.1.9.5.2 Configuring IPv4 IS-IS to Import External Routes ... 628
1.1.9.5.3 Verifying the Configuration of Interaction Between IPv4 IS-IS and Other Routing Protocols ... 631
1.1.9.6 Adjusting the IPv4 IS-IS Route Convergence Speed ... 631
1.1.9.6.1 Configuring the Interval for Detecting IS-IS Neighboring Device Failures ... 632
1.1.9.6.2 Setting Flooding Parameters of SNPs and LSPs ... 633
1.1.9.6.3 Setting the SPF Calculation Interval ... 638
1.1.9.6.4 Configuring Convergence Priorities for IPv4 IS-IS Routes ... 640
1.1.9.6.5 Enabling IS-IS Intelligent Convergence ... 640
1.1.9.6.6 Verifying the IPv4 IS-IS Route Convergence Speed Configuration ... 641
1.1.9.7 Configuring an IS-IS Multi-Instance Process ... 641
1.1.9.8 Enabling the Advertisement of IPv4 Delay Information ... 643
1.1.9.9 Configuring an IS-IS Process to Advertise IPv4 Packet Loss Rates ... 645
1.1.9.10 Configuring Static BFD for IS-IS ... 646
1.1.9.11 Configuring Dynamic BFD for IS-IS ... 648
1.1.9.11.1 Configuring BFD Globally ... 648
1.1.9.11.2 Configuring BFD for an IS-IS Process ... 648
1.1.9.11.3 (Optional) Configuring BFD on a Specified Interface ... 650
1.1.9.11.4 Verifying the Configuration of Dynamic BFD for IS-IS ... 651
1.1.9.12 Configuring Track BFD for IS-IS ... 652
1.1.9.13 Configuring IS-IS IPv4 MT to Isolate Multicast Services from Unicast Services ... 653
1.1.9.13.1 Enabling IPv4 MT for an IS-IS Process ... 654
1.1.9.13.2 Enabling MT for IPv4 an IS-IS Interface ... 655
1.1.9.13.3 Verifying the Configuration of IS-IS IPv4 Multi-Topologies Used to Isolate Multicast Services from Unicast Services ... 656
1.1.9.14 Configuring IPv4 IS-IS Route Summarization ... 656

1.1.9.15 Configuring IS-IS Auto FRR....658
1.1.9.16 Configuring Basic IPv6 IS-IS Functions....660
1.1.9.16.1 Creating an IPv6 IS-IS Process....661
1.1.9.16.2 Configuring an IPv6 IS-IS Interface....663
1.1.9.16.3 (Optional) Configuring a Cost for IS-IS Interfaces (IPv6)....665
1.1.9.16.4 (Optional) Configuring IPv6 IS-IS Attributes for Interfaces on Networks of Different Types....668
1.1.9.16.5 (Optional) Configuring IS-IS to Adjust the Flooding Rate (IPv6)....670
1.1.9.16.6 (Optional) Enabling LSP fragment extension on an IPv6 IS-IS device....671
1.1.9.16.7 Verifying the Basic IPv6 IS-IS Configuration....671
1.1.9.17 Configuring IPv6 IS-IS Route Selection....672
1.1.9.17.1 Configuring IPv6 IS-IS Route Leaking....673
1.1.9.17.2 Configuring Rules for Selecting Equal-Cost IPv6 IS-IS Routes....676
1.1.9.17.3 Controlling Whether to Add IS-IS Routes to the IPv6 Routing Table....677
1.1.9.17.4 Configuring an Overload Bit for an IPv6 IS-IS Device....679
1.1.9.17.5 Configuring IS-IS to Generate IPv6 Default Routes....679
1.1.9.17.6 Configuring an IPv6 IS-IS Interface to Automatically Adjust the Link Cost....681
1.1.9.17.7 Verifying the IPv6 IS-IS Route Selection Configuration....683
1.1.9.18 Configuring IPv6 IS-IS to Interact with Other Routing Protocols....683
1.1.9.18.1 Configuring a Priority Value for IPv6 IS-IS....684
1.1.9.18.2 Configuring IPv6 IS-IS to Import External Routes....685
1.1.9.18.3 Verifying the Configuration....688
1.1.9.19 Adjusting the IPv6 IS-IS Route Convergence Speed....688
1.1.9.19.1 Configuring the Interval for Detecting IS-IS Neighboring Device Failures....689
1.1.9.19.2 Setting Flooding Parameters of SNPs and LSPs....690
1.1.9.19.3 Setting the SPF Calculation Interval....695
1.1.9.19.4 Configuring Convergence Priorities for IPv6 IS-IS Routes....696
1.1.9.19.5 Configuring Interface Address Advertisement Suppression....697
1.1.9.19.6 Verifying the IPv6 IS-IS Route Convergence Speed Configuration....697
1.1.9.20 Enabling the Advertisement of IPv6 Delay Information....698
1.1.9.21 Configuring an IS-IS Process to Advertise IPv6 Packet Loss Rates....699
1.1.9.22 Configuring IS-IS Neighbor Relationship Flapping Suppression....701
1.1.9.23 Disabling IS-IS Interface Flapping Suppression....703
1.1.9.24 Disabling Routing Loop Detection for Routes Imported to IS-IS....703
1.1.9.25 Configuring IS-IS Purge Source Tracing....704
1.1.9.26 Configuring Dynamic BFD for IPv6 IS-IS....705
1.1.9.26.1 Configuring BFD Globally....706
1.1.9.26.2 Configuring BFD for an IS-IS IPv6 Process....706
1.1.9.26.3 (Optional) Configuring BFD on a Specified IPv6 Interface....708
1.1.9.26.4 Verifying the Configuration....709
1.1.9.27 Configuring Track BFD for IPv6 IS-IS....709
1.1.9.28 Configuring IS-IS IPv6 MT to Isolate Multicast Services from Unicast Services....710
1.1.9.28.1 Enabling MT for an IPv6 IS-IS Process....711

1.1.9.28.2 Enabling MT for an IPv6 IS-IS Interface....712
1.1.9.28.3 Verifying the Configuration of IS-IS IPv6 Multi-Topologies Used to Isolate Multicast Services from Unicast Services....713
1.1.9.29 Configuring IS-IS MT to Isolate IPv4 Services from IPv6 Services....714
1.1.9.29.1 Enabling MT for an IS-IS Process....714
1.1.9.29.2 Enabling MT for an IS-IS Interface....716
1.1.9.29.3 Verifying the Configuration of IS-IS Multi-Topologies Used to Isolate IPv4 Services from IPv6 Services....717
1.1.9.30 Configuring IPv6 IS-IS Route Summarization....718
1.1.9.31 Configuring IPv6 IS-IS Auto FRR....719
1.1.9.32 Configuring an IS-IS Link Group....722
1.1.9.33 Configuring IS-IS Local MT....723
1.1.9.33.1 Enabling Local MT....724
1.1.9.33.2 (Optional) Controlling the Size of the MIGP Routing Table....725
1.1.9.33.3 Verifying the Configuration of IS-IS Local MT....726
1.1.9.34 Improving IS-IS Network Security....727
1.1.9.34.1 Configuring IS-IS Authentication....727
1.1.9.34.2 Configuring the Optional Checksum....731
1.1.9.34.3 Verifying the Configuration of Improving IS-IS Network Security....732
1.1.9.35 Configuring Whitelist Session-CAR for IS-IS....732
1.1.9.36 Configuring Micro-Isolation CAR for IS-IS....733
1.1.9.37 Maintaining IS-IS....734
1.1.9.37.1 Resetting IS-IS....734
1.1.9.37.2 Suppressing IS-IS....735
1.1.9.37.3 Disabling IS-IS Memory Overload Control....735
1.1.9.37.4 Disabling IS-IS CPU Overload Control....736
1.1.9.37.5 (Optional) Configuring IS-IS to Discard a Specified LSP....737
1.1.9.38 IS-IS Configuration Examples....737
1.1.9.38.1 Example for Configuring Basic IS-IS Functions....738
1.1.9.38.2 Example for Configuring IS-IS DIS Election....743
1.1.9.38.3 Example for Configuring IS-IS Load Balancing....749
1.1.9.38.4 Example for Configuring IS-IS to Interact with BGP....753
1.1.9.38.5 Example for Configuring IS-IS Multi-Instance Processes....757
1.1.9.38.6 Example for Configuring Static BFD for IS-IS....762
1.1.9.38.7 Example for Configuring Dynamic BFD for IS-IS....767
1.1.9.38.8 Example for Configuring Dynamic BFD for IPv6 IS-IS....773
1.1.9.38.9 Example for Configuring Basic IS-IS IPv6 Functions....780
1.1.9.38.10 Example for Configuring IS-IS Auto FRR....785
1.1.9.38.11 Example for Configuring IS-IS MT....790
1.1.9.38.12 Example for Configuring Routing Loop Detection for Routes Imported from OSPF to IS-IS....797
1.1.9.38.13 Example for Configuring IS-IS Route Summarization....806
1.1.9.38.14 Example for Configuring Local MT....810
1.1.9.38.15 Example for Configuring IPv4 IS-IS Route Recursion to IPv6 Next Hops....818

1.1.10 BGP Configuration....820
1.1.10.1 BGP Overview....821
1.1.10.2 Configuration Precautions for BGP....823
1.1.10.3 Configuring Basic BGP Functions....833
1.1.10.3.1 Starting a BGP Process....834
1.1.10.3.2 Configuring a BGP Peer....835
1.1.10.3.3 Configuring BGP to Import Routes....838
1.1.10.3.4 Verifying the Configuration....840
1.1.10.4 Configuring the Format of BGP 4-Byte AS Numbers....841
1.1.10.5 Configuring BGP Route Attributes....842
1.1.10.5.1 Setting the BGP Preference....842
1.1.10.5.2 Setting a PrefVal for BGP Routes....843
1.1.10.5.3 Setting the Default Local_Pref Attribute for the Local Device....844
1.1.10.5.4 Configuring MED Attributes for BGP Routes....845
1.1.10.5.5 Configuring the Next_Hop Attribute....848
1.1.10.5.6 Setting the AS_Path Attribute....853
1.1.10.5.7 Configuring AIGP Attributes for Routes....857
1.1.10.5.8 Configuring Attr_Set Attribute Encapsulation or Parsing....859
1.1.10.5.9 Verifying the BGP Route Attribute Configuration....859
1.1.10.6 Configuring BGP Routing Policies....860
1.1.10.6.1 Configuring BGP Filters....861
1.1.10.6.2 Applying a Policy to BGP Route Advertisement....872
1.1.10.6.3 Applying a Policy to BGP Route Acceptance....876
1.1.10.6.4 Configuring BGP Soft Reset....880
1.1.10.6.5 Verifying the Configuration....881
1.1.10.7 Configuring BGP XPL....882
1.1.10.7.1 Using XPL to Filter the BGP Routes to Be Advertised....882
1.1.10.7.2 Using XPL to Filter the BGP Routes to Be Received....884
1.1.10.8 Configuring BGP Route Summarization....885
1.1.10.9 Configuring BGP to Generate a Summary Default Route....887
1.1.10.10 Configuring a BGP Peer Group....889
1.1.10.10.1 Creating an IBGP Peer Group....889
1.1.10.10.2 Creating a Pure EBGP Peer Group....890
1.1.10.10.3 Creating a Mixed EBGP Peer Group....891
1.1.10.10.4 Verifying the BGP Peer Group Configuration....892
1.1.10.11 Configuring a BGP Route Reflector....892
1.1.10.11.1 Configuring a Route Reflector and Specifying Clients....893
1.1.10.11.2 (Optional) Disabling Route Reflection Between Clients Through the RR....894
1.1.10.11.3 (Optional) Setting the Cluster ID of a Route Reflector....895
1.1.10.11.4 (Optional) Preventing a Device from Adding BGP Routes to the IP Routing Table....896
1.1.10.11.5 (Optional) Enabling an RR to Modify Route Attributes Using an Export Policy....897
1.1.10.11.6 Verifying the BGP Route Reflector Configuration....898

1.1.10.12 Configuring a BGP Confederation....898
1.1.10.13 Configuring BGP Community Attributes....900
1.1.10.13.1 Configuring a Community Attribute-Related Policy....900
1.1.10.13.2 Configuring Community Attribute Advertisement....901
1.1.10.13.3 Verifying the BGP Community Attribute Configuration....903
1.1.10.14 Configuring the BGP Large-Community Attribute....904
1.1.10.14.1 Configuring a Route-Policy Related to the Large-Community Attribute....904
1.1.10.14.2 Configuring the Device to Advertise the Large-Community Attribute....904
1.1.10.14.3 Verifying the Large-Community Configuration....905
1.1.10.15 Configuring Prefix-based BGP ORF....905
1.1.10.16 Adjusting the BGP Network Convergence Speed....906
1.1.10.16.1 Configuring BGP Keepalive and Hold Timers....908
1.1.10.16.2 Configuring a MinRouteAdvertisementIntervalTimer....910
1.1.10.16.3 Disabling Fast Reset of EBGP Connections....910
1.1.10.16.4 Configuring a BGP ConnectRetry Timer....911
1.1.10.16.5 Enabling Slow Peer Detection....912
1.1.10.16.6 Verifying the Configuration of the BGP Network Convergence Speed....913
1.1.10.17 Configuring a Dynamic BGP Peer Group....914
1.1.10.18 Configuring a BGP Device to Send a Default Route to Its Peer....915
1.1.10.19 Configuring a Device to Advertise BGP Supernet Unicast Routes to BGP Peers....917
1.1.10.20 Configuring BGP Load Balancing....918
1.1.10.21 Configuring BGP LSP Load Balancing....925
1.1.10.22 Configuring a BGP SR LSP....926
1.1.10.23 Configuring Path MTU Auto Discovery....930
1.1.10.24 Configuring BGP Route Recursion to the Default Route....933
1.1.10.25 Configuring BGP Next Hop Recursion Based on a Route-Policy....934
1.1.10.26 Configuring AIGP value on a Route-Policy....935
1.1.10.27 Configuring the POPGO Function....939
1.1.10.28 Configuring conversion from BGP IPv4 Unicast Routes to Labeled Routes....940
1.1.10.29 Configuring BFD for BGP....942
1.1.10.30 Configuring BGP Peer Tracking....947
1.1.10.31 Configuring BGP Auto FRR....948
1.1.10.32 Configuring Delayed Response to BGP Next Hop Recursion Changes....949
1.1.10.33 Setting a Specified Peer or Each Peer in a Peer Group as an Independent Update Peer-Group 952
1.1.10.34 Configuring a Delay in Releasing Obtained Labels in a BGP LSP FRR Switchover Scenario....953
1.1.10.35 Configuring the Route Server Function....955
1.1.10.36 Configuring the BGP GR Helper....956
1.1.10.36.1 Enabling BGP GR....956
1.1.10.36.2 Configuring the Parameter for a BGP GR Session....957
1.1.10.36.3 Verifying the BGP GR Helper Configuration....957
1.1.10.37 Enabling GR for BGP Peers....957
1.1.10.38 Configuring BGP Best-external....959

1.1.10.39 Configuring BGP Add-Path 960
1.1.10.40 Configuring BMP 963
1.1.10.41 Configuring BGP Route Dampening 968
1.1.10.42 Configuring Suppression on BGP Peer Flapping 969
1.1.10.43 Configuring Flapping Suppression Involved in BGP Next Hop Recursion 970
1.1.10.44 Configuring BGP-LS 972
1.1.10.45 Configuring the Entropy Label Capability for a BGP LSP 975
1.1.10.46 Configuring BGP RPD 977
1.1.10.47 Configuring IBGP Peers to Establish MPLS Local IFNET Tunnels 979
1.1.10.48 Configuring the YANG Management Mode of BGP 980
1.1.10.49 Improving BGP Security 982
1.1.10.49.1 Configuring MD5 Authentication 983
1.1.10.49.2 Configuring Keychain Authentication 985
1.1.10.49.3 Configuring TCP-AO Authentication 985
1.1.10.49.4 Configuring BGP GTSM 986
1.1.10.49.5 Configuring ROA 988
1.1.10.49.6 Configuring Regional Validation 992
1.1.10.49.7 Enabling SSL/TLS Authentication for BGP 994
1.1.10.50 Configuring BGP Extensions 995
1.1.10.51 Configuring BGP Multi-Instance 995
1.1.10.52 Maintaining BGP 996
1.1.10.52.1 Resetting BGP Connections 996
1.1.10.52.2 Clearing BGP Statistics 997
1.1.10.52.3 Configuring BGP to Record Peer Status Changes and Event Information 997
1.1.10.52.4 Disabling the PAF Restriction for the Feature Indicating Whether the Number of Received BGP Routes Exceeds the Upper Limit 998
1.1.10.52.5 Configuring a Mode in Which BGP Path Attributes Are Processed 999
1.1.10.52.6 Setting a Priority That Determines the Disconnection Order of a BGP Peer Relationship If Memory Overload Occurs 1000
1.1.10.52.7 Enabling BGP to Discard New Routes If a Memory Exception Occurs 1001
1.1.10.52.8 Configuring Whitelist Session-CAR for BGP 1002
1.1.10.52.9 Configuring Whitelist Session-CAR for BMP 1003
1.1.10.52.10 Configuring Whitelist Session-CAR for RPKI 1004
1.1.10.52.11 Configuring Micro-Segment CAR for BGP 1005
1.1.10.53 BGP Route Selection Rules 1005
1.1.10.53.1 Route Processing on the BGP Router 1006
1.1.10.53.2 Route Selection Rules 1007
1.1.10.53.3 BGP Routing Table 1012
1.1.10.53.4 Route Attributes 1018
1.1.10.53.5 (Optional) Configuring BGP to Ignore the Reachability of the Next Hops of Received BGP VPNv4 Routes 1081
1.1.10.53.6 Configuring BGP to Preferentially Select the Routes with Next Hops Recursing to SRv6 TE Policies 1082

1.1.10.53.7 Configuring BGP to Preferentially Select the Prefix Routes That Are Learned from VPNv4, VPNv6, or EVPN Peers and Are Leaked to a VPN Instance 1084
1.1.10.54 Configuration Examples for BGP 1085
1.1.10.54.1 Example for Configuring Basic BGP Functions 1085
1.1.10.54.2 Example for Configuring BGP to Interact with an IGP 1093
1.1.10.54.3 Example for Configuring the MED Attribute to Control Route Selection 1097
1.1.10.54.4 Example for Configuring an AS_Path Filter 1101
1.1.10.54.5 Example for Configuring BGP RRs 1106
1.1.10.54.6 Example for Configuring a BGP Confederation 1111
1.1.10.54.7 Example for Configuring the BGP Community Attribute 1117
1.1.10.54.8 Example for Configuring Prefix-based BGP ORF 1120
1.1.10.54.9 Example for Configuring BGP Route Dampening 1127
1.1.10.54.10 Example for Configuring BGP Default Route Advertisement 1131
1.1.10.54.11 Example for Configuring BGP Load Balancing 1139
1.1.10.54.12 Example for Configuring BGP Next Hop Recursion Based on a Routing Policy 1144
1.1.10.54.13 Example for Configuring Routing Policies to Control BGP Route Advertisement and Acceptance 1150
1.1.10.54.14 Example for Configuring BFD for BGP 1155
1.1.10.54.15 Example for Configuring BGP Auto FRR 1161
1.1.10.54.16 Example for Configuring BGP ADD-PATH 1167
1.1.10.54.17 Example for Configuring BGP Keychain Authentication 1174
1.1.10.54.18 Example for Configuring BGP-LS 1179
1.1.10.54.19 Example for Configuring BGP RPD 1186
1.1.10.54.20 Example for Configuring Dynamic BGP Peer Groups 1191
1.1.10.54.21 Example for Configuring BGP Multi-Instance 1195
1.1.10.54.22 Example for Configuring a BGP SR LSP 1201
1.1.10.54.23 Example for Configuring a BGP Virtual Link 1213
1.1.10.54.24 Example for Configuring BGP Routing Loop Detection 1236
1.1.11 BGP4+ Configuration 1243
1.1.11.1 Overview of BGP4+ 1243
1.1.11.2 Configuration Precautions for BGP4+ 1244
1.1.11.3 Configuring Basic BGP4+ Functions 1245
1.1.11.3.1 Starting a BGP Process 1245
1.1.11.3.2 Configuring IPv6 Peers 1246
1.1.11.3.3 (Optional) Configuring the Local Interface for a BGP4+ Connection 1249
1.1.11.3.4 Verifying the Configuration 1250
1.1.11.4 Configuring a BGP4+ Peer Group 1250
1.1.11.5 Controlling Route Import 1253
1.1.11.5.1 Configuring BGP4+ to Import Routes 1253
1.1.11.5.2 Configuring BGP4+ to Import Default Routes 1255
1.1.11.5.3 Verifying the Configuration 1255
1.1.11.6 Controlling Route Advertisement 1255
1.1.11.6.1 Configuring BGP4+ Route Summarization 1256

1.1.11.6.2 Configuring BGP4+ to Advertise Default Routes to Peers or peer groups.....1257
1.1.11.6.3 Enabling a BGP4+ Device to Add the Community Attribute to Routes to Be Advertised.....1258
1.1.11.6.4 Enabling a BGP4+ Device to Add the Large-Community Attribute to Routes to Be Advertised 1259
1.1.11.6.5 Setting the Interval at Which Update Packets Are Sent.....1260
1.1.11.6.6 Verifying the Configuration.....1260
1.1.11.7 Controlling BGP4+ Route Selection.....1261
1.1.11.7.1 Setting a BGP4+ Preference.....1261
1.1.11.7.2 Setting a PrefVal for BGP4+ Routes.....1262
1.1.11.7.3 Setting the Default Local_Pref Attribute for the Local Device.....1262
1.1.11.7.4 Setting the MED Attribute.....1263
1.1.11.7.5 Setting the Next_Hop Attribute.....1264
1.1.11.7.6 Setting the AS_Path Attribute.....1268
1.1.11.7.7 Configuring AIGP Attributes for Routes.....1271
1.1.11.7.8 Verifying the Configuration.....1272
1.1.11.8 Configuring BGP4+ Routing Policies.....1273
1.1.11.8.1 Configuring BGP4+ Filters.....1273
1.1.11.8.2 Configuring a Route-Policy.....1280
1.1.11.8.3 Applying a Policy to BGP4+ Route Advertisement.....1283
1.1.11.8.4 Configuring a Policy for Receiving BGP4+ Routes.....1286
1.1.11.8.5 Configuring BGP4+ Soft Reset.....1289
1.1.11.8.6 Verifying the Configuration.....1290
1.1.11.9 Configuring BGP4+ XPL.....1291
1.1.11.9.1 Using XPL to Filter the BGP4+ Routes to Be Advertised.....1291
1.1.11.9.2 Using XPL to Filter the BGP4+ Routes to Be Received.....1292
1.1.11.10 Configuring BGP4+ Route Recursion to the Default Route.....1293
1.1.11.11 Configuring BGP4+ to Generate Locator Routes.....1294
1.1.11.12 Configuring BGP4+ Load Balancing.....1295
1.1.11.13 Configuring BGP4+ to Generate a Summary Default Route.....1301
1.1.11.14 Configuring BGP4+ Connection Parameters.....1302
1.1.11.14.1 Configuring Timers for BGP4+ Peers.....1303
1.1.11.14.2 Enabling Fast Reset of EBGP Connections.....1304
1.1.11.14.3 Configuring a BGP4+ ConnectRetry Timer.....1305
1.1.11.14.4 Verifying the Configuration.....1306
1.1.11.15 Configuring BGP4+ RRs.....1306
1.1.11.15.1 Configuring an RR and Specifying Its Clients.....1307
1.1.11.15.2 (Optional) Disabling Route Reflection Between Clients Through the RR.....1308
1.1.11.15.3 (Optional) Configuring a Cluster ID for RRs.....1309
1.1.11.15.4 (Optional) Preventing BGP4+ from Adding Routes to the IP Routing Table.....1310
1.1.11.15.5 (Optional) Enabling an RR to Modify Route Attributes Using an Export Policy.....1310
1.1.11.15.6 Verifying the Configuration.....1312
1.1.11.16 Configuring a BGP4+ Confederation.....1312
1.1.11.17 Configuring BGP4+ Community Attributes.....1313

1.1.11.17.1 Configuring a Community Attribute-Related Policy.....1314
1.1.11.17.2 Configuring Community Attribute Advertisement.....1315
1.1.11.17.3 Verifying the Configuration.....1317
1.1.11.18 Configuring BFD for BGP4+.....1317
1.1.11.19 Configuring BGP4+ Auto FRR.....1319
1.1.11.20 Setting a Specified BGP4+ Peer or Each Peer in a Peer Group as an Independent Update Peer-Group.....1320
1.1.11.21 Configuring the Route Server Function.....1321
1.1.11.22 Configuring the BGP4+ GR Helper.....1322
1.1.11.22.1 Enabling BGP4+ GR.....1322
1.1.11.22.2 Configuring BGP4+ GR Session Parameters.....1323
1.1.11.22.3 Verifying the Configuration.....1323
1.1.11.23 Configuring the BGP4+ Add-Path Function.....1324
1.1.11.24 Configuring BMP.....1325
1.1.11.25 Enabling GR for BGP4+ Peers.....1329
1.1.11.26 Configuring BGP4+ Route Dampening.....1330
1.1.11.27 Configuring Suppression on BGP4+ Peer Flapping.....1332
1.1.11.28 Configuring Flapping Suppression Involved in BGP4+ Next Hop Recursion.....1333
1.1.11.29 Configuring BGP-LS (IPv6).....1334
1.1.11.30 Configuring BGP4+ Security.....1338
1.1.11.30.1 Configuring BGP4+ Authentication.....1339
1.1.11.30.2 Configuring BGP4+ GTSM.....1341
1.1.11.30.3 Configuring ROA.....1343
1.1.11.30.4 Configuring Regional Validation.....1347
1.1.11.31 Configuring BGP4+ 6PE.....1349
1.1.11.31.1 Configuring the IPv4/IPv6 Dual Stack.....1349
1.1.11.31.2 Configuring an LDP LSP on an IPv4 Network.....1350
1.1.11.31.3 Establishing a 6PE Peer Relationship Between PEs.....1351
1.1.11.31.4 (Optional) Enabling 6PE Routes Sharing the Explicit Null Label.....1352
1.1.11.31.5 Configuring Route Exchange Between a PE and a CE.....1353
1.1.11.31.6 Verifying the Configuration.....1353
1.1.11.32 Enabling the Device to Recurse BGP IPv6 Unicast Routes to LSPs.....1353
1.1.11.33 Maintaining BGP4+.....1355
1.1.11.33.1 Resetting BGP4+ Connections.....1355
1.1.11.33.2 Clearing BGP4+ Statistics.....1356
1.1.11.33.3 Configuring a Mode in Which BGP4+ Path Attributes Are Processed.....1356
1.1.11.33.4 Setting a Priority That Determines the Disconnection Order of a BGP4+ Peer Relationship If Memory Overload Occurs.....1357
1.1.11.33.5 Enabling BGP4+ to Discard New Routes in the Case of a Memory Exception.....1358
1.1.11.33.6 Configuring Whitelist Session-CAR for BGP4+.....1359
1.1.11.33.7 Configuring Whitelist Session-CAR for BMP.....1360
1.1.11.33.8 Configuring Whitelist Session-CAR for RPKI.....1361
1.1.11.34 Configuration Examples for BGP4+.....1362

1.1.11.34.1 Example for Configuring Basic BGP4+ Functions.... 1362
1.1.11.34.2 Example for Configuring BGP4+ Route Reflection.... 1367
1.1.11.34.3 Example for Configuring BGP4+ Keychain.... 1371
1.1.11.34.4 Example for Configuring BFD for BGP4+.... 1376
1.1.11.34.5 Example for Configuring BGP4+ 6PE.... 1382
1.1.12 Routing Policy Configuration.... 1389
1.1.12.1 Overview of Routing Policies.... 1389
1.1.12.2 Configuration Precautions for Routing Policy.... 1391
1.1.12.3 Configuring an IP Prefix List.... 1391
1.1.12.3.1 Configuring an IPv4 Prefix List.... 1391
1.1.12.3.2 Configuring an IPv6 Prefix List.... 1392
1.1.12.3.3 Verifying the Configuration.... 1393
1.1.12.4 Configuring an ACL.... 1393
1.1.12.5 Configuring an AS_Path Filter.... 1395
1.1.12.6 Configuring a Community Filter.... 1396
1.1.12.7 Configuring a Large-Community Filter.... 1397
1.1.12.8 Configuring an Extcommunity Filter.... 1398
1.1.12.9 Configuring an RD Filter.... 1400
1.1.12.10 Configuring a Route-Policy.... 1400
1.1.12.10.1 Creating a Route-Policy.... 1401
1.1.12.10.2 (Optional) Configuring an if-match Clause.... 1402
1.1.12.10.3 (Optional) Configuring an **apply** Clause.... 1407
1.1.12.10.4 (Optional) Further Matching Routes Against a Specified Node.... 1409
1.1.12.10.5 Applying a Route-Policy.... 1409
1.1.12.10.6 Verifying the Route-Policy Configuration.... 1430
1.1.12.11 Applying Filters to Received Routes.... 1431
1.1.12.11.1 Configuring RIP to Filter the Received Routes.... 1431
1.1.12.11.2 Configuring OSPF to Filter the Received Routes.... 1433
1.1.12.11.3 Configuring IS-IS to Filter the Received Routes.... 1435
1.1.12.11.4 Verifying the Configuration of Applying Filters to Received Routes.... 1436
1.1.12.12 Applying Filters to Routes to Be Advertised.... 1437
1.1.12.12.1 Configuring RIP to Filter the Routes to Be Advertised.... 1438
1.1.12.12.2 Configuring OSPF to Filter the Routes to Be Advertised.... 1439
1.1.12.12.3 Configuring IS-IS to Filter the Routes to Be Advertised.... 1441
1.1.12.12.4 Verifying the Configuration of Applying Filters to the Advertised Routes.... 1442
1.1.12.13 Applying Filters to Imported Routes.... 1443
1.1.12.13.1 Configuring RIP to Import External Routes.... 1444
1.1.12.13.2 Configuring OSPF to Import External Routes.... 1444
1.1.12.13.3 Configuring IS-IS to Import External Routes.... 1446
1.1.12.13.4 Verifying the Configuration of Applying Filters to the Imported Routes.... 1447
1.1.12.14 Configuration Examples for Routing Policies.... 1447
1.1.12.14.1 Example for Filtering the Received Routes and Those to Be Advertised.... 1447

1.1.12.14.2 Example for Applying a Route-Policy to Imported Routes....1452
1.1.12.14.3 Example for Applying Route-Policies....1457
1.1.12.14.4 Example for Configuring a Basic or Advanced VPN-Target Extended Community Filter....1475
1.1.13 XPL Configuration....1482
1.1.13.1 Overview of XPL....1482
1.1.13.2 Configuration Precautions for XPL....1489
1.1.13.3 Configuring a Global Variable Set....1489
1.1.13.4 Configuring a Route Attribute Set....1491
1.1.13.4.1 Configuring an IPv4 Prefix Set....1492
1.1.13.4.2 Configuring an IPv6 Prefix Set....1495
1.1.13.4.3 Configuring an AS_Path Set....1497
1.1.13.4.4 Configuring a Community Set....1499
1.1.13.4.5 Configuring Large-Community Sets....1502
1.1.13.4.6 Configuring an RD Set....1505
1.1.13.4.7 Configuring a Route Target Set....1508
1.1.13.4.8 Configuring an SoO Set....1511
1.1.13.4.9 Verifying the Route Attribute Set Configuration....1514
1.1.13.5 Configuring a Route-Filter....1515
1.1.13.6 Applying Route-Filters....1519
1.1.13.7 Configuration Examples for XPL....1522
1.1.13.7.1 Example for Using XPL to Filter the Routes to Be Received and Advertised (Paragraph-by-Paragraph Editing)....1522
1.1.13.7.2 Example for Using XPL to Filter the Routes to Be Received and Advertised (Line-by-Line Editing)....1530
1.1.13.8 XPL Paragraph Editing Clauses....1538
1.1.13.8.1 Common Clauses....1538
1.1.13.8.2 Condition Clauses....1539
1.1.13.8.3 Action Clauses....1592
1.1.14 Route Monitoring Group Configuration....1630
1.1.14.1 Overview of Route Monitoring Groups....1630
1.1.14.2 Configuration Precautions for Route Monitoring Group....1630
1.1.14.3 Configuring a Route Monitoring Group....1630
1.1.14.4 Configuration Examples for Route Monitoring Groups....1634
1.1.14.4.1 Example for Configuring Association Between a Route Monitoring Group and a Dual-Device Hot Backup Module....1634

Figures

Figure 1-1 IP ring network.....11
Figure 1-2 IP ring network with IPv4 FRR poison reverse enabled.....11
Figure 1-3 IP ring network.....14
Figure 1-4 IP ring network with IPv6 FRR poison reverse enabled.....14
Figure 1-5 Networking for the IPv4 direct route responding to L3VE interface status changes after a delay.....21
Figure 1-6 Networking of an IPv6 direct route responding to L3VE interface status changes after a delay.....23
Figure 1-7 Networking for the association between the IPv4 direct route and PW status.....26
Figure 1-8 Networking of the association between the IPv6 direct route and PW status.....28
Figure 1-9 Networking for an L2VPN accessing an L3VPN.....30
Figure 1-10 Networking diagram of advertising IPv4 ARP Vlink direct routes on the public network.....33
Figure 1-11 Networking diagram of advertising IPv6 NDP Vlink direct routes on the public network.....35
Figure 1-12 Coexistence of EVPN dual-homing single-active and L3VPN.....37
Figure 1-13 Networking for configuring public network IPv4 FRR.....45
Figure 1-14 Networking for configuring public network IPv6 FRR.....50
Figure 1-15 Association between direct routes and a VRRP group.....54
Figure 1-16 Networking diagram of importing IPv4 ARP Vlink direct routes to BGP.....59
Figure 1-17 Networking diagram of importing IPv6 NDP Vlink direct routes to BGP4+.....66
Figure 1-18 Networking for association between LDP and static routes.....86
Figure 1-19 Networking for configuring IPv4 static routes.....97
Figure 1-20 Networking for configuring IPv4 floating static routes.....100
Figure 1-21 Configuring BFD for static routes.....104
Figure 1-22 Configuring static BFD for IPv4 static routes.....107
Figure 1-23 NQA for IPv4 static routes.....110
Figure 1-24 Associating EFM with IPv4 static routes.....118
Figure 1-25 Networking for configuring FRR for IPv4 static routes on the public network.....121
Figure 1-26 Networking for configuring association between LDP and static routes.....127
Figure 1-27 FRR for IPv6 static routes.....145
Figure 1-28 Networking for configuring IPv6 static routes.....147
Figure 1-29 Networking for configuring IPv6 floating static routes.....150
Figure 1-30 Networking for configuring dynamic BFD for IPv6 static routes.....154

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-31 Configuring static BFD for IPv6 static routes.....158
Figure 1-32 NQA for IPv6 static routes.....162
Figure 1-33 Networking for configuring FRR for IPv6 static routes on the public network.....168
Figure 1-34 Networking diagram of equal-cost routes.....200
Figure 1-35 Networking diagram of suppressing an interface from receiving and sending OSPF packets.....203
Figure 1-36 OSPF IP FRR link protection.....256
Figure 1-37 OSPF FRR in the scenario where multiple nodes advertise the same route.....257
Figure 1-38 Solution of selecting a backup path for OSPF IP FRR.....259
Figure 1-39 Maximum-cost (set by LDP-IGP synchronization) path first solution.....260
Figure 1-40 Networking for configuring basic OSPF functions.....285
Figure 1-41 Networking for configuring an OSPF virtual links.....292
Figure 1-42 Networking for configuring an OSPF stub area.....296
Figure 1-43 Networking for configuring an OSPF NSSA.....302
Figure 1-44 Networking for configuring OSPF DR election.....307
Figure 1-45 Networking for configuring OSPF load balancing.....312
Figure 1-46 Networking for configuring OSPF fast convergence.....317
Figure 1-47 Configuring local OSPF MT.....322
Figure 1-48 Networking for configuring OSPF IP FRR.....330
Figure 1-49 Networking for configuring BFD for OSPF.....335
Figure 1-50 Networking for configuring OSPF-BGP synchronization.....340
Figure 1-51 Routing loop detection for routes imported from BGP to OSPF.....349
Figure 1-52 Networking for configuring OSPF multi-area adjacency.....356
Figure 1-53 Networking for configuring an OSPF sham link.....362
Figure 1-54 Flowchart for configuring OSPFv3 attributes on different types of networks.....381
Figure 1-55 OSPFv3 application in the VPN Option A scenario.....402
Figure 1-56 OSPFv3 FRR in the scenario where multiple nodes advertise the same route.....414
Figure 1-57 Solution of selecting a backup path for OSPFv3 IP FRR.....415
Figure 1-58 Networking for configuring OSPFv3 areas.....449
Figure 1-59 Networking for configuring OSPFv3 DR election.....453
Figure 1-60 Networking for configuring OSPFv3 IP FRR.....458
Figure 1-61 Networking for configuring BFD for OSPFv3.....464
Figure 1-62 IPsec for OSPFv3.....468
Figure 1-63 Example for configuring OSPFv3 route summarization on an ASBR.....473
Figure 1-64 Example for configuring OSPFv3 route summarization on an ABR.....479
Figure 1-65 Networking for configuring an OSPFv3 NSSA.....486
Figure 1-66 Routing loop detection for OSPFv3 multi-process mutual route import.....492
Figure 1-67 Disabling RIP-2 classful summarization.....510
Figure 1-68 Scenario where an interface is disabled from sending packets.....520
Figure 1-69 Scenario where an interface is disabled from receiving packets.....521
Figure 1-70 Networking diagram of unaffiliated static BFD for RIP.....531
Figure 1-71 RIP neighbor relationship over a P2P link.....534
Figure 1-72 RIP neighbor relationship over a broadcast link.....536

Figure 1-73 Configuring basic RIP functions.....544
Figure 1-74 Network diagram of configuring routing loop prevention.....548
Figure 1-75 Network diagram of configuring RIP to import external routes.....553
Figure 1-76 Configuring BFD for RIP.....557
Figure 1-77 Configuring static BFD for RIP.....562
Figure 1-78 Network diagram of basic RIPng function configuration.....589
Figure 1-79 Network diagram of configuring routing loop prevention.....594
Figure 1-80 Configuring basic IS-IS functions.....738
Figure 1-81 Configuring IS-IS DIS election.....744
Figure 1-82 Configuring IS-IS load balancing.....749
Figure 1-83 Configuring IS-IS to interact with BGP.....753
Figure 1-84 Networking for configuring IS-IS multi-instance processes.....758
Figure 1-85 Configuring static BFD for IS-IS.....763
Figure 1-86 Configuring dynamic BFD for IS-IS.....767
Figure 1-87 Configuring dynamic BFD for IPv6 IS-IS.....774
Figure 1-88 Configuring basic IS-IS IPv6 functions.....780
Figure 1-89 Configuring IS-IS Auto FRR.....786
Figure 1-90 Configuring IS-IS MT.....791
Figure 1-91 Routing loop detection for routes imported from OSPF to IS-IS.....797
Figure 1-92 IS-IS route summarization networking.....807
Figure 1-93 Local IS-IS MT networking.....811
Figure 1-94 Networking diagram for configuring IS-IS route recursion to IPv6 next hops.....818
Figure 1-95 BGP networking.....822
Figure 1-96 Inter-AS VPN Option C networking with RRs deployed.....850
Figure 1-97 Network diagram of Layer 2 transparent transmission.....852
Figure 1-98 Typical networking.....888
Figure 1-99 Typical networking with hierarchical RRs.....893
Figure 1-100 Configuring a device to send a default route to its peer.....915
Figure 1-101 Load balancing among VPN unicast routes and leaked routes.....923
Figure 1-102 Configuring a BGP SR LSP.....927
Figure 1-103 Network without path MTU auto discovery.....930
Figure 1-104 Network with path MTU auto discovery.....931
Figure 1-105 AIGP application networking.....936
Figure 1-106 POPGO networking.....939
Figure 1-107 Networking for configuring conversion from BGP IPv4 unicast routes to labeled routes.....940
Figure 1-108 Networking diagram for a traffic loop.....941
Figure 1-109 Networking diagram for configuring the BGP next hop recursion change delayed response.....950
Figure 1-110 Networking diagram for a critical recursion change.....950
Figure 1-111 Networking diagram for a non-critical recursion change.....951
Figure 1-112 BGP LSP FRR switchover networking.....954
Figure 1-113 Networking with master and backup PEs.....960

Figure 1-114 Networking for configuring BGP Add-Path....961
Figure 1-115 Load balancing performed on transit nodes.... 976
Figure 1-116 Typical networking of inbound traffic optimization.... 977
Figure 1-117 Route processing on the BGP Router....1006
Figure 1-118 BGP route selection process.... 1008
Figure 1-119 BGP route unreachability.... 1023
Figure 1-120 PreVal application networking....1026
Figure 1-121 Local_Pref application networking.... 1031
Figure 1-122 Local_Pref-based BGP route selection networking (1)....1037
Figure 1-123 Local_Pref-based BGP route selection networking (2)....1041
Figure 1-124 Networking.... 1044
Figure 1-125 AIGP application networking....1047
Figure 1-126 Networking in which a private AS needs to be deleted....1052
Figure 1-127 Networking in which new AS numbers are added to the AS_Path....1053
Figure 1-128 Networking diagram with Origin configurations.... 1058
Figure 1-129 Networking diagram for MED applications....1064
Figure 1-130 Networking diagram for MED applications....1066
Figure 1-131 Peer type application networking....1071
Figure 1-132 Networking diagram with IGP cost configurations....1072
Figure 1-133 Networking diagram with Cluster_List configurations.... 1075
Figure 1-134 Networking diagram with Originator_ID configurations....1077
Figure 1-135 Networking in which two links are used to establish two BGP peer relationships. 1080
Figure 1-136 Configuring BGP to ignore the reachability of the next hops of received BGP VPNv4 routes....1082
Figure 1-137 Network diagram of configuring BGP to preferentially select the routes with next hops recursing to SRv6 TE Policies.... 1083
Figure 1-138 Configuring basic BGP functions.... 1086
Figure 1-139 Configuring BGP to interact with an IGP....1093
Figure 1-140 Configuring the MED attribute to control route selection....1097
Figure 1-141 Configuring BGP route filtering.... 1102
Figure 1-142 Configuring BGP RRs.... 1107
Figure 1-143 Configuring the confederation.... 1112
Figure 1-144 Configuring the BGP community attribute....1117
Figure 1-145 Networking diagram for configuring prefix-based BGP ORF....1121
Figure 1-146 Configuring BGP route dampening.... 1127
Figure 1-147 Configuring BGP default route advertisement.... 1131
Figure 1-148 Configuring BGP load balancing.... 1140
Figure 1-149 Networking diagram for configuring BGP next hop recursion based on a routing policy.... 1145
Figure 1-150 Network diagram of filtering routes to be advertised and accepted....1150
Figure 1-151 Configuring BFD for BGP.... 1156
Figure 1-152 Configuring BGP Auto FRR.... 1161
Figure 1-153 Networking for configuring BGP ADD-PATH....1167

Figure 1-154 Networking diagram of configuring BGP keychain authentication.....1174
Figure 1-155 Configuring BGP-LS.....1179
Figure 1-156 Networking for configuring BGP RPD.....1187
Figure 1-157 Network diagram of configuring dynamic BGP peer groups.....1191
Figure 1-158 BGP multi-instance networking.....1196
Figure 1-159 Configuring a BGP SR LSP.....1202
Figure 1-160 Configuring a BGP virtual link.....1213
Figure 1-161 Configuring BGP routing loop detection.....1237
Figure 1-162 BGP networking.....1244
Figure 1-163 Inter-AS VPN Option C networking with RRs deployed.....1266
Figure 1-164 Typical networking.....1295
Figure 1-165 Load balancing among VPN unicast routes and leaked routes.....1300
Figure 1-166 Typical networking.....1302
Figure 1-167 Networking with hierarchical RRs.....1307
Figure 1-168 Networking with 6PE.....1349
Figure 1-169 Scenario where BGP IPv6 unicast routes recurse to LSPs.....1354
Figure 1-170 Configuring basic BGP4+ functions.....1362
Figure 1-171 Configuring BGP4+ route reflection.....1367
Figure 1-172 Networking diagram for configuring BGP4+ keychain.....1372
Figure 1-173 Configuring BFD for BGP4+.....1376
Figure 1-174 Configuring BGP4+ 6PE.....1382
Figure 1-175 Filtering the received routes and those to be advertised.....1448
Figure 1-176 Applying a route-policy to imported routes.....1453
Figure 1-177 Applying route-policies.....1457
Figure 1-178 Networking for configuring a basic or advanced VPN-target extended community filter.....1476
Figure 1-179 Cyclic figure for filtering the routes to be received and advertised.....1523
Figure 1-180 Cyclic figure for filtering the routes to be received and advertised.....1531
Figure 1-181 Route monitoring group.....1632
Figure 1-182 Configuring association between a route monitoring group and a dual-device hot backup module.....1634

Tables

Table 1-1 Feature requirements.... 6
Table 1-2 Feature requirements.... 76
Table 1-3 Interface IPv4 addresses.... 112
Table 1-4 NQA item values.... 112
Table 1-5 NQA item values.... 163
Table 1-6 Differences among RIP, OSPF, and IS-IS.... 176
Table 1-7 Feature requirements.... 178
Table 1-8 Network types supported by OSPF.... 192
Table 1-9 Default route advertising mode.... 209
Table 1-10 OSPF features supported by OSPF multi-area adjacency interfaces.... 274
Table 1-11 IP address of each interface.... 323
Table 1-12 Feature requirements.... 373
Table 1-13 Feature requirements.... 500
Table 1-14 Feature requirements.... 598
Table 1-15 Mapping between IS-IS interface costs and interface bandwidth.... 609
Table 1-16 Mapping between IS-IS interface costs and interface bandwidth.... 667
Table 1-17 Feature requirements.... 823
Table 1-18 Description of metacharacters.... 863
Table 1-19 Attribute comparison of the routes learned from Device A and Device B.... 938
Table 1-20 Configuration modes for the **compatible** and **single-hop-prefer** parameters.... 944
Table 1-21 Key points for route processing.... 1006
Table 1-22 BGP route selection process.... 1009
Table 1-23 Commands used to check route attributes.... 1012
Table 1-24 Description of the **display bgp routing-table** command output.... 1013
Table 1-25 Description of the **display bgp routing-table** command output.... 1015
Table 1-26 Next_Hop processing.... 1019
Table 1-27 Unreachable next hop.... 1023
Table 1-28 Methods to modify the PreVal value.... 1025
Table 1-29 Route attribute comparison of the route learned from ISP1 and that learned from ISP2.... 1027
Table 1-30 Methods to modify the Local_Pref value.... 1030
Table 1-31 Route attribute comparison of the routes learned from ISP1 and Device B.... 1032
Table 1-32 Route attribute comparison of the routes learned from Device A and Device B.... 1033
Table 1-33 Route attribute comparison of the routes learned from Device A and ISP2.... 1036

 Copyright © Huawei Technologies Co., Ltd.

Table 1-34 Route attribute comparison of the routes learned from ISP1 and Device B.....1039
Table 1-35 Attribute comparison of the routes learned from Device A and Device B.....1050
Table 1-36 AS_Path-based route selection rules.....1051
Table 1-37 Attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC.1055
Table 1-38 Attribute comparison of the routes learned from Device C and Device D.....1060
Table 1-39 Attribute comparison of the routes learned from Device C and Device D.....1061
Table 1-40 Methods to modify the MED value.....1062
Table 1-41 Attribute comparison of the routes that DeviceB learns from DeviceC and DeviceD.1065
Table 1-42 Attribute comparison of the routes that DeviceB learns from DeviceC and DeviceD.1068
Table 1-43 Attribute comparison of the routes learned from Device A and Device B.....1073
Table 1-44 Attribute comparison of the routes learned by Device A from Device B and Device C 1076
Table 1-45 Attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC.1078
Table 1-46 IP addresses of the interfaces.....1107
Table 1-47 Configurations of each device.....1168
Table 1-48 Description of metacharacters.....1274
Table 1-49 Differences between the routing policy and PBR.....1390
Table 1-50 Feature requirements.....1391
Table 1-51 Applying a route-policy to direct routes.....1410
Table 1-52 Applying a route-policy to RIP routes.....1412
Table 1-53 Applying a route-policy to RIPng routes.....1413
Table 1-54 Applying a route-policy to IPv4 IS-IS routes in the IS-IS view.....1415
Table 1-55 Applying a route-policy to IPv6 IS-IS routes in the IS-IS view.....1418
Table 1-56 Applying a route-policy to OSPF routes.....1420
Table 1-57 Applying a route-policy to OSPFv3 routes.....1422
Table 1-58 Applying a route-policy to BGP routes.....1424
Table 1-59 Applying a route-policy to BGP4+ routes.....1427
Table 1-60 IP addresses of physical interfaces.....1458
Table 1-61 IP addresses of loopback interfaces.....1459
Table 1-62 BGP parameter values.....1460
Table 1-63 Comparison between XPL and routing policies.....1482
Table 1-64 Line-by-line and paragraph-by-paragraph editing comparison.....1483
Table 1-65 Shortcut keys commonly used in the paragraph editing interface.....1484
Table 1-66 Feature requirements.....1489
Table 1-67 Route-filter application in OSPF.....1520
Table 1-68 Route-filter application in RIP.....1520
Table 1-69 Route-filter application in RIPng.....1521
Table 1-70 Route-filter application in IS-IS.....1521
Table 1-71 Route-filter application in BGP.....1521
Table 1-72 Route-filter application in BGP4+.....1522

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 About This Document

Purpose

This document provides the basic concepts, configuration procedures, and configuration examples in different application scenarios of the IP Routing feature.

Licensing Requirements

For details about the License, see the License Guide.

Related Version

The following table lists the product version related to this document.

Product Name	Version
HUAWEI NE40E-M2 series	V800R022C00SPC600
iMaster NCE-IP	V100R022C00SPC100

Intended Audience

This document is intended for:

- Data configuration engineers

 Copyright © Huawei Technologies Co., Ltd.

- Commissioning engineers
- Network monitoring engineers
- System maintenance engineers

Security Declaration

- Notice on Limited Command Permission

  The documentation describes commands when you use Huawei devices and make network deployment and maintenance. The interfaces and commands for production, manufacturing, repair for returned products are not described here.

  If some advanced commands and compatible commands for engineering or fault location are incorrectly used, exceptions may occur or services may be interrupted. It is recommended that the advanced commands be used by engineers with high rights. If necessary, you can apply to Huawei for the permissions to use advanced commands.
- Encryption algorithm declaration

  The encryption algorithms DES/3DES/RSA (with a key length of less than 2048 bits)/MD5 (in digital signature scenarios and password encryption)/SHA1 (in digital signature scenarios) have a low security, which may bring security risks. If protocols allowed, using more secure encryption algorithms, such as AES/RSA (with a key length of at least 2048 bits)/SHA2/HMAC-SHA2 is recommended.

  For security purposes, insecure protocols Telnet, FTP, and TFTP as well as weak security algorithms in BGP, LDP, PECP, MSDP, DCN, TCP-AO, MSTP, VRRP, E-Trunk, AAA, IPsec, BFD, QX, SSH, SNMP, IS-IS, RIP, SSL, NTP, OSPF, and keychain features are not recommended. To use such weak security algorithms, run the undo crypto weak-algorithm disable command to enable the weak security algorithm function. For details, see the Configuration Guide.
- Password configuration declaration
  - When the password encryption mode is cipher, avoid setting both the start and end characters of a password to "%^%#". This causes the password to be displayed directly in the configuration file.
  - To further improve device security, periodically change the password.
- Personal data declaration
  - Your purchased products, services, or features may use users' some personal data during service operation or fault locating. You must define user privacy policies in compliance with local laws and take proper measures to fully protect personal data.
  - When discarding, recycling, or reusing a device, back up or delete data on the device as required to prevent data leakage. If you need support, contact after-sales technical support personnel.
- Feature declaration
  - The NetStream feature may be used to analyze the communication information of terminal customers for network traffic statistics and management purposes. Before enabling the NetStream feature, ensure that it is performed within the boundaries permitted by applicable laws and regulations. Effective measures must be taken to ensure that information is securely protected.

 Copyright © Huawei Technologies Co., Ltd.

  - The mirroring feature may be used to analyze the communication information of terminal customers for a maintenance purpose. Before enabling the mirroring function, ensure that it is performed within the boundaries permitted by applicable laws and regulations. Effective measures must be taken to ensure that information is securely protected.
  - The packet header obtaining feature may be used to collect or store some communication information about specific customers for transmission fault and error detection purposes. Huawei cannot offer services to collect or store this information unilaterally. Before enabling the function, ensure that it is performed within the boundaries permitted by applicable laws and regulations. Effective measures must be taken to ensure that information is securely protected.
- Reliability design declaration

  Network planning and site design must comply with reliability design principles and provide device- and solution-level protection. Device-level protection includes planning principles of dual-network and inter-board dual-link to avoid single point or single link of failure. Solution-level protection refers to a fast convergence mechanism, such as FRR and VRRP. If solution-level protection is used, ensure that the primary and backup paths do not share links or transmission devices. Otherwise, solution-level protection may fail to take effect.

Special Declaration

- This document serves only as a guide. The content is written based on device information gathered under lab conditions. The content provided by this document is intended to be taken as general guidance, and does not cover all scenarios. The content provided by this document may be different from the information on user device interfaces due to factors such as version upgrades and differences in device models, board restrictions, and configuration files. The actual user device information takes precedence over the content provided by this document. The preceding differences are beyond the scope of this document.
- The maximum values provided in this document are obtained in specific lab environments (for example, only a certain type of board or protocol is configured on a tested device). The actually obtained maximum values may be different from the maximum values provided in this document due to factors such as differences in hardware configurations and carried services.
- Interface numbers used in this document are examples. Use the existing interface numbers on devices for configuration.
- The pictures of hardware in this document are for reference only.
- The supported boards are described in the document. Whether a customization requirement can be met is subject to the information provided at the pre-sales interface.
- In this document, public IP addresses may be used in feature introduction and configuration examples and are for reference only unless otherwise specified.
- The configuration precautions described in this document may not accurately reflect all scenarios.
- Log Reference and Alarm Reference respectively describe the logs and alarms for which a trigger mechanism is available. The actual logs and alarms that the product can generate depend on the types of services it supports.

 Copyright © Huawei Technologies Co., Ltd.

Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
DANGER	Indicates a hazard with a high level of risk which, if not avoided, will result in death or serious injury.
WARNING	Indicates a hazard with a medium level of risk which, if not avoided, could result in death or serious injury.
CAUTION	Indicates a hazard with a low level of risk which, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a potentially hazardous situation which, if not avoided, could result in equipment damage, data loss, performance deterioration, or unanticipated results. NOTICE is used to address practices not related to personal injury.
NOTE	Supplements the important information in the main text. NOTE is used to address information not related to personal injury, equipment damage, and environment deterioration.

Command Conventions

The command conventions that may be found in this document are defined as follows.

Convention	Description
Boldface	The keywords of a command line are in **boldface**.
Italic	Command arguments are in *italics*.
[ ]	Items (keywords or arguments) in brackets [ ] are optional.
{ x | y | ... }	Optional items are grouped in braces and separated by vertical bars. One item is selected.
[x | y | ...]	Optional items are grouped in brackets and separated by vertical bars. One item is selected or no item is selected.

 Copyright © Huawei Technologies Co., Ltd.

Convention	Description
{ x | y | ... }*	Optional items are grouped in braces and separated by vertical bars. A minimum of one item or a maximum of all items can be selected.
[x | y | ...]*	Optional items are grouped in brackets and separated by vertical bars. Several items or no item can be selected.
&<1-n>	The parameter before the & sign can be repeated 1 to n times.
#	A line starting with the # sign is comments.

Change History

Changes between document issues are cumulative. The latest document issue contains all the changes made in earlier issues.

- **Changes in Issue 01 (2022-10-31)**

  This issue is the first official release. The software version of this issue is V800R022C00SPC600.

1.1.2 Basic IP Routing Configuration

IP routing is the basic concept on data communication networks.

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.

 Copyright © Huawei Technologies Co., Ltd.

1.1.2.2 Configuration Precautions for Basic IP Routing

Feature Requirements

Table 1-1 Feature requirements

Feature Requirements	Series	Models
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. Plan services properly.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

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.

1.1.2.4 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.

1.1.2.5 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.

1.1.2.6 Configuring IPv4 FRR

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

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.2.6.1 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**

 Copyright © Huawei Technologies Co., Ltd.

The configuration is committed.

----End

1.1.2.6.2 (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 1-1**, 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 1-2**.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-1 IP ring network

Figure 1-2 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

1.1.2.6.3 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

1.1.2.7 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.2.7.1 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.

NOTE

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

1.1.2.7.2 (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 1-3**, 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 1-4**.

Figure 1-3 IP ring network

Figure 1-4 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

1.1.2.7.3 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

1.1.2.8 Configuring Route Recursion to the Default Route

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

Usage Scenario

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

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.

1.1.2.9 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.

1.1.2.10 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 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.

1.1.2.10.1 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. The association allows the VRRP-enabled devices to modify the cost of each direct route to the virtual IP network segment based on the VRRP status.

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

1.1.2.10.2 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

1.1.2.10.3 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 lows through the master device in the VRRP group.

----End

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 1-5**, 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 1-5 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.

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 **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.

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 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.

1.1.2.12 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 1-6**, 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 1-6 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 **dircet-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.

1.1.2.13 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 1-7**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-7 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.

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 **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.

1.1.2.14 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 1-8**, 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 1-8 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.

 Copyright © Huawei Technologies Co., Ltd.

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 **dircet-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.

1.1.2.15 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 1-9 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 1-9 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 to the BGP routing table.

NOTE

After a static route is imported to 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 a specified VPN instance and allowed to inherit the cost of the route to which the static route recurses.

vpn-source-name specifies the VPN instance that is bound to the L3VE sub-interface, and *nexthop-address* specifies the address of the base station's interface connected 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.

1.1.2.16 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 1-10**, 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 1-10 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.

1.1.2.17 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 1-11**, 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 1-11 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.

1.1.2.18 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 1-12**, 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 1-12 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.

1.1.2.19 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.

1.1.2.20 Maintaining IP Routes

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

1.1.2.20.1 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

1.1.2.20.2 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

1.1.2.20.3 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

1.1.2.20.4 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

1.1.2.20.5 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

1.1.2.20.6 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.

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

1.1.2.21 Configuration Examples for IP Routing Basis

This section describes several configuration examples.

1.1.2.21.1 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 1-13**, 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 1-13 Networking for configuring public network IPv4 FRR

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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. Configure basic OSPF functions on Device T, Device A, and Device C (OSPFv3 process ID is 1).
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_
         Destinations : 1        Routes : 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: GigabitEthernet0/2/0
     TunnelID: 0x0                Flags: D
```

```
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: GigabitEthernet0/3/0
    TunnelID: 0x0                Flags: 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_
         Destinations : 1        Routes : 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: 0x40000041
RelayNextHop: 0.0.0.0          Interface: GigabitEthernet0/2/0
    TunnelID: 0x0                Flags: D
   BkNextHop: 192.168.20.2   BkInterface: GigabitEthernet0/3/0
     BkLabel: NULL            SecTunnelID: 0x0
BkPETunnelID: 0x0        BkPESecTunnelID: 0x0
BkIndirectID: 0x80000081

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: 0x80000081
RelayNextHop: 0.0.0.0          Interface: GigabitEthernet0/3/0
    TunnelID: 0x0                Flags: 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 0/2/0
[~DeviceT-GigabitEthernet0/2/0] shutdown
[*DeviceT-GigabitEthernet0/2/0] commit
[~DeviceT-GigabitEthernet0/2/0] 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_
         Destinations : 1        Routes : 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: 0x80000081
 RelayNextHop: 0.0.0.0           Interface: GigabitEthernet0/3/0
     TunnelID: 0x0                   Flags: D
```

The preceding command output shows that traffic has been switched to the backup link (Link B).

----End

Configuration Files

- Device T configuration file

```
#
sysname DeviceT
#
ip frr
#
isis 1
 network-entity 10.0000.0000.0001.00
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.10.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.20.1 255.255.255.0
 isis enable 1
#
ospf 1
 area 0.0.0.0
  network 192.168.10.0 0.0.0.255
 area 0.0.0.1
  network 172.16.1.0 0.0.0.255
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.10.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.11.2 255.255.255.0
#
ospf 1
 area 0.0.0.0
```

```
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.20.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.17.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.11.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.21.1 255.255.255.0
 isis enable 1
#
ospf 1
 area 0.0.0.0
  network 192.168.11.0 0.0.0.255
  network 192.168.21.0 0.0.0.255
 area 0.0.0.2
  network 172.17.1.0 0.0.0.255
#
return
```

1.1.2.21.2 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 1-14**, 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 1-14 Networking for configuring public network IPv6 FRR

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 0/2/0  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 0/3/0  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 0/2/0  Flags        : D
BkNextHop    : 2001:db8:100::2        BkInterface  : gigabitethernet 0/3/0
BkLabel      : NULL                   BkTunnelID   : 0x0
```

```
BkPETunnelID : 0x0                          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 0/3/0        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 0/2/0
[~DeviceT-GigabitEthernet0/2/0] shutdown
[*DeviceT-GigabitEthernet0/2/0] commit
[~DeviceT-GigabitEthernet0/2/0] 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 0/3/0        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 0.0.0.0
 area 0.0.0.1
#
interface GigabitEthernet0/1/0
 undo shutdown
```

```
 ipv6 enable
 ipv6 address 2001:db8:10::1/64
 ospfv3 1 area 0.0.0.1
 isis ipv6 enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:200::1/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/3/0
 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 0.0.0.0
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:200::2/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:201::2/64
 ospfv3 1 area 0.0.0.0
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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:100::2/64
 isis ipv6 enable 1
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
 area 0.0.0.2
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:20::1/64
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:201::1/64
 ospfv3 1 area 0.0.0.2
#
interface GigabitEthernet0/3/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:101::1/64
 isis enable 1
 isis ipv6 enable 1
#
return
```

1.1.2.21.3 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 1-15**, 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 1-15 Association between direct routes and a VRRP group

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/1/1, 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 are 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 0/1/0 and GE 0/1/1 to VLAN 10 so that the Switch transparently transmits VRRP packets sent by Device 1 or Device 2.

```
<Switch> system-view
[~Switch] interface gigabitethernet 0/1/0
[~Switch-GigabitEthernet0/1/0] portswitch
[*Switch-GigabitEthernet0/1/0] commit
[*Switch-GigabitEthernet0/1/0] quit
[*Switch] interface gigabitethernet 0/1/1
[~Switch-GigabitEthernet0/1/1] portswitch
[*Switch-GigabitEthernet0/1/1] commit
[~Switch-GigabitEthernet0/1/1] quit
[*Switch] vlan 10
[*Switch-vlan10] port gigabitethernet 0/1/0
[*Switch-vlan10] port gigabitethernet 0/1/1
[*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 0/1/0
[*Device1-GigabitEthernet0/1/0] vrrp vrid 1 virtual-ip 10.1.1.111
[*Device1-GigabitEthernet0/1/0] vrrp vrid 1 priority 120
[*Device1-GigabitEthernet0/1/0] commit
[~Device1-GigabitEthernet0/1/0] quit
```

Create VRRP group 1 on Device 2

```
<Device2> system-view
[~Device2] interface gigabitethernet 0/1/0
[*Device2-GigabitEthernet0/1/0] vrrp vrid 1 virtual-ip 10.1.1.111
[*Device2-GigabitEthernet0/1/0] commit
[~Device2-GigabitEthernet0/1/0] 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-0.0.0.0] network 10.1.2.0 0.0.0.255
[*Device3-ospf-1-area-0.0.0.0] network 10.1.3.0 0.0.0.255
[*Device3-ospf-1-area-0.0.0.0] commit
[~Device3-ospf-1-area-0.0.0.0] quit
[~Device3-ospf-1] quit
```

Step 4 Associate direct routes with the VRRP group.

Configure Device1.

```
[~Device1] interface gigabitethernet 0/1/0
[*Device1-GigabitEthernet0/1/0] direct-route track vrrp vrid 1 degrade-cost 300
[*Device1-GigabitEthernet0/1/0] commit
[~Device1-GigabitEthernet0/1/0] quit
```

Configure Device2.

```
[~Device2] interface gigabitethernet 0/1/0
[*Device2-GigabitEthernet0/1/0] direct-route track vrrp vrid 1 degrade-cost 300
[*Device2-GigabitEthernet0/1/0] commit
[~Device2-GigabitEthernet0/1/0] 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        GigabitEthernet0/1/0
       10.1.1.2/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.1.1.111/32  O_ASE  150  0           D   10.1.2.2        GigabitEthernet0/1/1
     10.1.1.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
       10.1.2.0/24  Direct 0    0           D   10.1.2.1        GigabitEthernet0/1/1
       10.1.2.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/1
     10.1.2.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/1
       10.1.3.0/24  OSPF   10   2           D   10.1.2.2        GigabitEthernet0/1/1
      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        GigabitEthernet0/1/0
```

----End

Configuration Files

- Switch configuration file

```
#
sysname Switch
#
vlan batch 10
#
interface GigabitEthernet0/1/0
 portswitch
 undo shutdown
 port default vlan 10
#
interface GigabitEthernet0/1/1
 portswitch
 undo shutdown
 port default vlan 10
#
```

- Device 1 configuration file

```
#
sysname Device1
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
#
ospf 1
 default cost inherit-metric
 import-route direct
 area 0.0.0.0
  network 10.1.3.0 0.0.0.255
#
```

- Device 2 configuration file

```
#
sysname Device2
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/1
 undo shutdown
```

```
 ip address 10.1.2.1 255.255.255.0
#
ospf 1
 default cost inherit-metric
 import-route direct
 area 0.0.0.0
  network 10.1.2.0 0.0.0.255
#
```

- Device 3 configuration file

```
#
sysname Device3
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.2.2 255.255.255.0
#
ospf 1
 area 0.0.0.0
  network 10.1.2.0 0.0.0.255
  network 10.1.3.0 0.0.0.255
#
```

1.1.2.21.4 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 1-16**, 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 1-16 Networking diagram of importing IPv4 ARP Vlink direct routes to BGP

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] undo shutdown
[*DeviceA-GigabitEthernet0/1/0] ip address 10.1.1.3 24
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure Device B.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[~DeviceB] interface GigabitEthernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] undo shutdown
[*DeviceB-GigabitEthernet0/1/0] ip address 10.1.1.4 24
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] quit
```

Configure Device C.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceC
[*HUAWEI] commit
[~DeviceC] interface GigabitEthernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] undo shutdown
[*DeviceC-GigabitEthernet0/2/0] ip address 10.2.1.1 24
[*DeviceC-GigabitEthernet0/2/0] commit
[~DeviceC-GigabitEthernet0/2/0] quit
```

Configure Device D.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceD
[*HUAWEI] commit
[~DeviceD] interface GigabitEthernet 0/1/0
[*DeviceD-GigabitEthernet0/1/0] undo shutdown
[*DeviceD-GigabitEthernet0/1/0] ip address 10.2.1.2 24
[*DeviceD-GigabitEthernet0/1/0] commit
[~DeviceD-GigabitEthernet0/1/0] 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 0/1/0
[*SwitchA-GigabitEthernet0/1/0] portswitch
[*SwitchA-GigabitEthernet0/1/0] undo shutdown
[*SwitchA-GigabitEthernet0/1/0] port link-type access
[*SwitchA-GigabitEthernet0/1/0] port default vlan 10
[*SwitchA-GigabitEthernet0/1/0] quit
[*SwitchA] interface GigabitEthernet 0/2/0
[*SwitchA-GigabitEthernet0/2/0] portswitch
[*SwitchA-GigabitEthernet0/2/0] undo shutdown
[*SwitchA-GigabitEthernet0/2/0] port link-type access
[*SwitchA-GigabitEthernet0/2/0] port default vlan 10
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*SwitchA-GigabitEthernet0/2/0] quit
[*SwitchA] interface GigabitEthernet 0/3/0
[*SwitchA-GigabitEthernet0/3/0] portswitch
[*SwitchA-GigabitEthernet0/3/0] undo shutdown
[*SwitchA-GigabitEthernet0/3/0] port link-type access
[*SwitchA-GigabitEthernet0/3/0] port default vlan 10
[*SwitchA-GigabitEthernet0/3/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] portswitch
[*DeviceC-GigabitEthernet0/1/0] undo shutdown
[*DeviceC-GigabitEthernet0/1/0] port link-type access
[*DeviceC-GigabitEthernet0/1/0] port default vlan 10
[*DeviceC-GigabitEthernet0/1/0] 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 GigabitEthernet0/1/0
 portswitch
 undo shutdown
 port link-type access
 port default vlan 10
#
interface GigabitEthernet0/2/0
 portswitch
 undo shutdown
```

```
 port link-type access
 port default vlan 10
#
interface GigabitEthernet0/3/0
 portswitch
 undo shutdown
 port link-type access
 port default vlan 10
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.3 255.255.255.0
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 portswitch
 undo shutdown
 port link-type access
 port default vlan 10
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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
```

1.1.2.21.5 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 1-17**, 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 1-17 Networking diagram of importing IPv6 NDP Vlink direct routes to BGP4+

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ipv6 enable
[*DeviceA-GigabitEthernet0/1/0] ipv6 address 2001:db8:2000::3 64
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure DeviceB.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[*DeviceB] interface GigabitEthernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] undo shutdown
[*DeviceB-GigabitEthernet0/1/0] ipv6 enable
[*DeviceB-GigabitEthernet0/1/0] ipv6 address 2001:db8:2000::4 64
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] quit
```

Configure DeviceC.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceC
[*HUAWEI] commit
[~DeviceC] interface GigabitEthernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ipv6 enable
[*DeviceC-GigabitEthernet0/2/0] ipv6 address 2001:db8:2001::1 64
[*DeviceC-GigabitEthernet0/2/0] commit
[~DeviceC-GigabitEthernet0/2/0] quit
```

Configure DeviceD.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceD
[*HUAWEI] commit
[~DeviceD] interface GigabitEthernet 0/1/0
[*DeviceD-GigabitEthernet0/1/0] ipv6 enable
[*DeviceD-GigabitEthernet0/1/0] ipv6 address 2001:db8:2001::2 64
[*DeviceD-GigabitEthernet0/1/0] commit
[~DeviceD-GigabitEthernet0/1/0] 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 0/1/0
[*SwitchA-GigabitEthernet0/1/0] portswitch
[*SwitchA-GigabitEthernet0/1/0] port link-type access
[*SwitchA-GigabitEthernet0/1/0] port default vlan 10
[*SwitchA-GigabitEthernet0/1/0] quit
[*SwitchA] interface GigabitEthernet 0/2/0
[*SwitchA-GigabitEthernet0/2/0] portswitch
[*SwitchA-GigabitEthernet0/2/0] port link-type access
[*SwitchA-GigabitEthernet0/2/0] port default vlan 10
[*SwitchA-GigabitEthernet0/2/0] quit
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*SwitchA] interface GigabitEthernet 0/3/0
[*SwitchA-GigabitEthernet0/3/0] portswitch
[*SwitchA-GigabitEthernet0/3/0] port link-type access
[*SwitchA-GigabitEthernet0/3/0] port default vlan 10
[*SwitchA-GigabitEthernet0/3/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] portswitch
[*DeviceC-GigabitEthernet0/1/0] port link-type access
[*DeviceC-GigabitEthernet0/1/0] port default vlan 10
[*DeviceC-GigabitEthernet0/1/0] 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 GigabitEthernet0/1/0
 portswitch
 undo shutdown
 port link-type access
 port default vlan 10
#
interface GigabitEthernet0/2/0
 portswitch
 undo shutdown
```

```
 port link-type access
 port default vlan 10
#
interface GigabitEthernet0/3/0
 portswitch
 undo shutdown
 port link-type access
 port default vlan 10
#
return
```

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2000::3/64
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 portswitch
 undo shutdown
 port link-type access
 port default vlan 10
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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
```

1.1.3 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.

1.1.3.1 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.

1.1.3.2 Configuration Precautions for IPv4 static route

Feature Requirements

Table 1-2 Feature requirements

Feature Requirements	Series	Models
The static routes to participate in FRR must have different priorities configured.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
Before you associate a static route with a BFD session, the session must have been created.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
When static routes are deployed, the following features are mutually exclusive: 1. BFD (Bidirectional Forwarding Detection) for static route 2. NQA (Network Quality Analysis) for static route 3. Association between static routes and EFM4 4. Permanent advertisement of static routes Determine which feature to use based on the actual network conditions: ■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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.3.3 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 NE40E 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 NE40E-M2 seriesUniversal Service 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.

1.1.3.3.1 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

1.1.3.3.2 (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

1.1.3.3.3 (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.

 Copyright © Huawei Technologies Co., Ltd.

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

1.1.3.3.4 (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

1.1.3.3.5 (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

1.1.3.3.6 (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

1.1.3.3.7 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

1.1.3.4 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.3.5 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 1-18**, 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 1-18 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* [*nexthop-address*] [**preference** *preference* | **tag** *tag*] * **ldp-sync** [**description** *text*]

     Association between LDP and static routes is configured.
  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.

1.1.3.6 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.

NOTE

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.

1.1.3.6.1 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 node.

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 none of min-rx-interval, min-tx-interval, or detect-multiplier is specified, the default values of the 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

Checking the Configurations

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.

1.1.3.6.2 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.
  a. Run **system-view**

     The system view is displayed.
  b. Run **bfd**

     BFD is globally enabled.
  c. Run **quit**

     The system view is displayed.
  d. 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.
  e. Run **discriminator local** *discr-value*

     The local discriminator of a static BFD session is set.
  f. 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 NE40E-M2 series Universal Service Router Configuration Guide - Network Reliability*.
  g. Run **commit**

     The configuration is committed.
- Associating an IPv4 Static Route with a BFD Session.
  a. 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

Checking the Configurations

Run the following commands to check the previous configuration.

- Run the **display bfd session** { **all** | **discriminator** *discr-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.

1.1.3.6.3 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.
  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 NE40E-M2 series Universal Service 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.
  b. 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* } [**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* } [**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.

1.1.3.6.4 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 the interface up or down event that is triggered by the change of the EFM OAM extension status and determines whether to activate static routes. This mechanism controls route advertisement and ensures that the traffic from the remote end can be correctly forwarded.

EFM is used on links between customer edges (CEs) and user-end provider edges (UPEs) on a metropolitan area network (MAN). It helps maintain the reliability and stability of connections between a user network and a carrier network. EFM

monitors and detects faults in P2P Ethernet physical links or simulated links. Using EFM on a user access network is recommended.

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 NE40E-M2 series Universal Service 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-num* 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.

1.1.3.7 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.

1.1.3.8 Configuration Examples for IPv4 Static Routes

This section provides configuration examples of IPv4 static routes.

1.1.3.8.1 Example for Configuring IPv4 Static Routes

You can configure IPv4 static routes to interconnect any two devices on an IPv4 network.

Networking Requirements

Figure 1-19 shows the IP addresses and masks of interfaces and hosts. It is required that any two hosts in **Figure 1-19** communicate through static routes.

Figure 1-19 Networking for configuring IPv4 static routes

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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        GigabitEthernet0/1/0
```

```
      10.1.1.0/24  Direct 0    0         D       10.1.1.1        GigabitEthernet0/2/0
      10.1.1.1/32  Direct 0    0         D       127.0.0.1       GigabitEthernet0/2/0
    10.1.1.255/32  Direct 0    0         D       127.0.0.1       GigabitEthernet0/2/0
      10.1.4.0/30  Direct 0    0         D       10.1.4.1        GigabitEthernet0/1/0
      10.1.4.1/32  Direct 0    0         D       127.0.0.1       GigabitEthernet0/1/0
      10.1.4.2/32  Direct 0    0         D       10.1.4.2        GigabitEthernet0/1/0
    10.1.4.255/32  Direct 0    0         D       127.0.0.1       GigabitEthernet0/1/0
     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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.252
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.2 255.255.255.252
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.4.5 255.255.255.252
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.6 255.255.255.252
#
interface GigabitEthernet0/2/0
 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
```

1.1.3.8.2 Example for Configuring IPv4 Floating Static Routes

IPv4 Floating static routes can be used for the static route backup.

Networking Requirements

Figure 1-20 shows the IPv4 addresses and masks of each Router interface and a host. 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 1-20 Networking for configuring IPv4 floating static routes

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/1, GE 0/1/2, and GE 0/1/3, 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 IPv4 static routes to 10.1.3.0/24 and 10.1.4.0/24.
4. On Device A, configure two IPv4 static routes to 10.1.5.0/24 with different priorities.
5. 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 10.1.3.0 24 10.1.1.2
[*DeviceA] ip route-static 10.1.4.0 24 10.1.2.2
[*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 : 3        Routes : 3        Configured Routes : 3
Static routing table status : <Active>
         Destinations : 3        Routes : 3

Destination/Mask    Proto  Pre  Cost      Flags NextHop         Interface

       10.1.3.0/24  Static 60   0           RD 10.1.1.2        GigabitEthernet0/1/1
       10.1.4.0/24  Static 60   0           RD 10.1.2.2        GigabitEthernet0/1/2
    10.1.5.0/24  Static 60   0           RD 10.1.1.2        GigabitEthernet0/1/1

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 0/1/1 of Device A to simulate a link fault.

```
[~DeviceA] interface GigabitEthernet 0/1/1
[~DeviceA-GigabitEthernet0/1/1] shutdown
[~DeviceA-GigabitEthernet0/1/1] 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-GigabitEthernet0/1/1] 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 : 2        Routes : 2        Configured Routes : 2
Static routing table status : <Active>
         Destinations : 2        Routes : 2

Destination/Mask    Proto  Pre  Cost      Flags NextHop         Interface

       10.1.4.0/24  Static 60   0           RD 10.1.2.2        GigabitEthernet0/1/2
    10.1.5.0/24  Static 100  0           RD 10.1.2.2        GigabitEthernet0/1/2

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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
ip route-static 10.1.3.0 24 10.1.1.2
ip route-static 10.1.4.0 24 10.1.2.2
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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/1/2
 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet0/1/2
 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet0/1/3
 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
```

 Copyright © Huawei Technologies Co., Ltd.

1.1.3.8.3 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 1-21**, 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 1-21 Configuring BFD for static routes

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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       GigabitEthernet0/1/0
      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       GigabitEthernet0/1/0
      10.10.1.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
      10.10.1.2/32  Direct 0    0           D   10.10.1.2       GigabitEthernet0/1/0
    10.10.1.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
```

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    : -
-------------------------------------------------------------------------------

   Total UP/DOWN Session Number : 1/0
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.10.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.10.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.3.8.4 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 1-22**, 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 1-22 Configuring static BFD for IPv4 static routes

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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         GigabitEthernet0/1/0
        1.1.1.0/24  Direct 0    0           D  1.1.1.1         GigabitEthernet0/1/0
        1.1.1.1/32  Direct 0    0           D  127.0.0.1        GigabitEthernet0/1/0
      1.1.1.255/32  Direct 0    0           D  127.0.0.1        GigabitEthernet0/1/0
255.255.255.255/32  Direct 0    0           D  127.0.0.1        InLoopBack0
```

Run the **shutdown** command on GE 0/1/0 of DeviceB to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] 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         GigabitEthernet0/1/0
        1.1.1.1/32  Direct 0    0           D  127.0.0.1        GigabitEthernet0/1/0
      1.1.1.255/32  Direct 0    0           D  127.0.0.1        GigabitEthernet0/1/0
255.255.255.255/32  Direct 0    0           D  127.0.0.1        InLoopBack0
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.3.8.5 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 1-23**, backup links are deployed on the IP metropolitan area network (MAN).

- Static routes are configured on DeviceB and DeviceC. DeviceB is the master device, while DeviceC is the backup 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 1-23 NQA for IPv4 static routes

NOTE

Interfaces 1 through 5 in this example represent GE 0/1/0, GE 0/1/1, GE 0/1/2, GE 0/1/3, and GE 0/2/3, respectively.

NOTE

In this networking, SwitchA and SwitchB provide 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.

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 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 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:

- Interface IPv4 addresses (for details, see **Table 1-3**)

Table 1-3 Interface IPv4 addresses

Device	Interface Number	IP Address
DeviceA	GE 0/1/0	172.16.3.2/24
	GE 0/2/3	172.16.4.1/24
DeviceB	GE 0/1/0	172.16.3.2/24
	GE 0/1/1	172.16.1.1/24
	GE 0/1/3	172.16.5.1/24
DeviceC	GE 0/1/0	172.16.6.1/24
	GE 0/1/2	172.16.2.1/24
	GE 0/2/3	172.16.2.1/24
SwitchA	VLANIF 10	172.16.1.2/24
	VLANIF 20	172.16.6.2/24
	VLANIF 30	172.16.7.1/24
SwitchB	VLANIF 10	172.16.2.2/24
	VLANIF 20	172.16.5.2/24
	VLANIF 30	172.16.8.1/24

- NQA item values (for details, see **Table 1-4**)

Table 1-4 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

 Copyright © Huawei Technologies Co., Ltd.

Item	Value
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 0/1/1 172.16.1.2 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 0/1/0 172.16.6.2
[*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 0/1/1 172.16.1.2 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      GigabitEthernet0/1/1
     172.16.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/1
   172.16.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/1
     172.16.3.0/24  Direct  0    0           D   172.16.3.2      GigabitEthernet0/1/0
     172.16.3.2/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.3.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     172.16.4.0/24  OSPF    10   2           D   172.16.3.1      GigabitEthernet0/1/0
     172.16.5.0/24  Direct  0    0           D   172.16.5.1      GigabitEthernet0/1/3
     172.16.5.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/3
   172.16.5.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/3
   172.16.7.0/24  Static  60   0           D   172.16.1.2      GigabitEthernet0/1/1
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      GigabitEthernet0/1/0
     172.16.3.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.3.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     172.16.4.0/24  Direct  0    0           D   172.16.4.1      GigabitEthernet0/2/3
     172.16.4.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/3
   172.16.4.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/3
     172.16.7.0/24  O_ASE   150  10          D   172.16.3.2      GigabitEthernet0/1/0
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 0/1/1 on DeviceB to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 0/1/1
[~DeviceB-GigabitEthernet0/1/1] shutdown
[~DeviceB-GigabitEthernet0/1/1] 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      GigabitEthernet0/1/0
     172.16.3.2/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.3.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     172.16.4.0/24  OSPF    10   2           D   172.16.3.1      GigabitEthernet0/1/0
     172.16.5.0/24  Direct  0    0           D   172.16.5.1      GigabitEthernet0/1/3
     172.16.5.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/3
   172.16.5.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/3
     172.16.7.0/24  O_ASE   150  20          D   172.16.3.1      GigabitEthernet0/1/0
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      GigabitEthernet0/1/0
     172.16.3.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.3.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     172.16.4.0/24  Direct  0    0           D   172.16.4.1      GigabitEthernet0/2/3
     172.16.4.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/3
   172.16.4.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/3
     172.16.7.0/24  O_ASE   150  20          D   172.16.4.2      GigabitEthernet0/2/3
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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.3.1 255.255.255.0
#
interface GigabitEthernet0/2/3
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.3.2 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet0/1/3
 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 GigabitEthernet0/1/1 172.16.1.2 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.6.1 255.255.255.0
#
```

 Copyright © Huawei Technologies Co., Ltd.

```
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 172.16.2.1 255.255.255.0
#
interface GigabitEthernet0/2/3
 undo shutdown
 ip address 172.16.4.2 255.255.255.0
#
ospf 1
 import-route static cost 20
 area 0.0.0.0
  network 172.16.4.0 0.0.0.255
#
ip route-static 172.16.7.0 255.255.255.0 GigabitEthernet0/1/0 172.16.6.2
#
return
```

1.1.3.8.6 Example for Associating EFM with IPv4 Static Routes

After EFM is associated with IPv4 static routes, the system responds to the interface Up or Down event that is triggered by the change of the EFM OAM extension status and determines whether to activate static routes. This mechanism controls route advertisement and ensures that the traffic from the remote end can be correctly forwarded.

Networking Requirements

In **Figure 1-24**, Device A and Device B are connected. EFM OAM is enabled on Device A and Device B. A static route destined for 1.1.1.1/32 is configured on Device A, and this static route is associated with EFM.

Figure 1-24 Associating EFM with IPv4 static routes

NOTE

Interfaces 1 and 2 in this example represent GE0/1/0 and GE0/1/1, 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] undo shutdown
[*DeviceA-GigabitEthernet0/1/0] efm enable
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
[~DeviceA] quit
```

Enable EFM OAM both globally and on the interface of Device B.

```
<DeviceB> system-view
[~DeviceB] efm enable
[*DeviceB] interface GigabitEthernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] undo shutdown
[*DeviceB-GigabitEthernet0/1/0] efm enable
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] quit
[~DeviceB] quit
```

Display EFM OAM session information on Device A.

```
<DeviceA> display efm session all
 Interface              EFM State              Loopback Timeout
 ---------------------------------------------------------------------
 GigabitEthernet0/1/0     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 GigabitEthernet0/1/1 172.16.1.2 track efm-state
GigabitEthernet0/1/0
[*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 GigabitEthernet0/1/1 172.16.1.2 track efm-state GigabitEthernet0/1/0
```

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      GigabitEthernet0/1/1
     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      GigabitEthernet0/1/1
    172.16.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/1
  172.16.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/1
255.255.255.255/32  Direct  0    0           D   127.0.0.1       InLoopBack0
```

Run the **shutdown** command on GE 0/1/0 of Device A to check whether the EFM OAM status changes.

```
<DeviceA> system-view
[~DeviceA] interface GigabitEthernet 0/1/0
[~DeviceA-GigabitEthernet0/1/0] shutdown
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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
  ---------------------------------------------------------------------
  GigabitEthernet0/1/0     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      GigabitEthernet0/1/1
    172.16.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/1
  172.16.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/1
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 GigabitEthernet0/1/0
 undo shutdown
 efm enable
#
interface GigabitEthernet0/1/1
 undo shutdown
```

```
 ip address 172.16.1.1 255.255.255.0
#
 ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet0/1/1 172.16.1.2 track efm-state
GigabitEthernet0/1/0
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
efm enable
#
interface GigabitEthernet0/1/0
 undo shutdown
 efm enable
#
interface GigabitEthernet0/1/1
 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
```

1.1.3.8.7 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 1-25**, 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 the Link B immediately.

Figure 1-25 Networking for configuring FRR for IPv4 static routes on the public network

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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. On Device T, set a higher priority for Link A to ensure that Link A becomes the primary link.
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 priorities of IPv4 static routes.

Procedure

Step 1 Configure an IP address for each interface. For configuration details, see **Configuration Files** in this section.

Step 2 Configure IPv4 static routes.

On Device A, configure IPv4 static routes.

```
[~DeviceA] ip route-static 172.16.1.0 24 GigabitEthernet0/1/0 192.168.10.1
[*DeviceA] ip route-static 172.17.1.0 24 GigabitEthernet0/2/0 192.168.11.1
[*DeviceA] commit
```

On Device B, configure static routes.

```
[~DeviceB] ip route-static 172.16.1.0 24 GigabitEthernet0/1/0 192.168.20.1
[*DeviceB] ip route-static 172.17.1.0 24 GigabitEthernet0/2/0 192.168.21.1
[*DeviceB] commit
```

On Device C, configure IPv4 static routes.

```
[~DeviceC] ip route-static 172.16.1.0 24 GigabitEthernet0/2/0 192.168.11.2
[*DeviceC] ip route-static 172.16.1.0 24 GigabitEthernet0/3/0 192.168.21.2
[*DeviceC] ip route-static 192.168.10.0 24 GigabitEthernet0/2/0 192.168.11.2
[*DeviceC] ip route-static 192.168.20.0 24 GigabitEthernet0/3/0 192.168.21.2
[*DeviceC] commit
```

On Device T, configure IPv4 static routes.

 Copyright © Huawei Technologies Co., Ltd.

```
[~DeviceT] ip route-static 172.17.1.0 24 GigabitEthernet0/2/0 192.168.10.2
[*DeviceT] ip route-static 172.17.1.0 24 GigabitEthernet0/3/0 192.168.20.2
[*DeviceT] ip route-static 192.168.11.0 24 GigabitEthernet0/2/0 192.168.10.2
[*DeviceT] ip route-static 192.168.21.0 24 GigabitEthernet0/3/0 192.168.20.2
[*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      GigabitEthernet0/1/0
     172.16.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.17.1.0/24  Static 60  0           D  192.168.10.2    GigabitEthernet0/2/0
                  Static 60  0           D  192.168.20.2    GigabitEthernet0/3/0
   192.168.10.0/24  Direct  0    0           D   192.168.10.1    GigabitEthernet0/2/0
   192.168.10.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
 192.168.10.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
   192.168.11.0/24  Static  60   0           D   192.168.10.2    GigabitEthernet0/2/0
   192.168.20.0/24  Direct  0    0           D   192.168.20.1    GigabitEthernet0/3/0
   192.168.20.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/3/0
 192.168.20.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/3/0
   192.168.21.0/24  Static  60   0           D   192.168.20.2    GigabitEthernet0/3/0
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 GigabitEthernet0/2/0 192.168.10.2  preference 40
[*DeviceT] commit
[*DeviceT] quit
```

Check the IP routing table of Device T. The following command output shows that the priorities of IPv4 static routes have 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      GigabitEthernet0/1/0
     172.16.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.17.1.0/24  Static 40  0           D  192.168.10.2    GigabitEthernet0/2/0
   192.168.10.0/24  Direct  0    0           D   192.168.10.1    GigabitEthernet0/2/0
   192.168.10.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
 192.168.10.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
   192.168.11.0/24  Static  60   0           D   192.168.10.2    GigabitEthernet0/2/0
   192.168.20.0/24  Direct  0    0           D   192.168.20.1    GigabitEthernet0/3/0
   192.168.20.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/3/0
```

```
 192.168.20.255/32  Direct  0    0          D   127.0.0.1       GigabitEthernet0/3/0
   192.168.21.0/24  Static  60   0          D   192.168.20.2    GigabitEthernet0/3/0
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: GigabitEthernet0/2/0
     TunnelID: 0x0                  Flags: D
    BkNextHop: 192.168.20.2   BkInterface: GigabitEthernet0/3/0
      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 GigabitEthernet0/2/0 192.168.10.2 preference 40 track
  bfd-session aa
  ```

Step 6 Simulate a fault on Link A.

```
<DeviceT> system-view
```

```
[~DeviceT] interface gigabitethernet 0/2/0
[~DeviceT-GigabitEthernet0/2/0] shutdown
[~DeviceT-GigabitEthernet0/2/0] commit
[~DeviceT-GigabitEthernet0/2/0] 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: GigabitEthernet0/3/0
     TunnelID: 0x0                  Flags: D
```

----End

Configuration Files

- Device T configuration file

```
#
sysname DeviceT
#
bfd
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.10.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/2/0 192.168.10.2 preference 40 track bfd-session aa
ip route-static 172.17.1.0 24 GigabitEthernet0/3/0 192.168.20.2
ip route-static 192.168.11.0 24 GigabitEthernet0/2/0 192.168.10.2
ip route-static 192.168.21.0 24 GigabitEthernet0/3/0 192.168.20.2
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.10.2 255.255.255.0
#
interface GigabitEthernet0/2/0
```

```
 undo shutdown
 ip address 192.168.11.2 255.255.255.0
#
ip route-static 172.16.1.0 24 GigabitEthernet0/1/0 192.168.10.1
ip route-static 172.17.1.0 24 GigabitEthernet0/2/0 192.168.11.1
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.20.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.21.2 255.255.255.0
#
ip route-static 172.16.1.0 24 GigabitEthernet0/1/0 192.168.20.1
ip route-static 172.17.1.0 24 GigabitEthernet0/2/0 192.168.10.1
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
bfd
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 172.17.1.0 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.11.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/2/0 192.168.11.2
ip route-static 172.16.1.0 24 GigabitEthernet0/3/0 192.168.21.2
ip route-static 192.168.10.0 255.255.255.0 GigabitEthernet0/2/0 192.168.11.2
ip route-static 192.168.20.0 255.255.255.0 GigabitEthernet0/3/0 192.168.21.2
#
return
```

1.1.3.8.8 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 1-26**, 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 A recovers from a fault.

Figure 1-26 Networking for configuring association between LDP and static routes

NOTE

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

Device Name	Interface Name	Interface IP Address	Device Name	Interface Name	Interface IP Address
LSRA	GE 0/1/0	10.1.1.1/30	LSRC	GE 0/1/0	10.2.1.2/30
	GE 0/2/0	10.2.1.1/30		GE 0/2/0	10.4.1.1/30
	Loopback0	1.1.1.1/32		Loopback0	3.3.3.3/32
LSRB	GE 0/1/0	10.1.1.2/30	LSRD	GE 0/1/0	10.3.1.2/30
	GE 0/2/0	10.3.1.1/30		GE 0/2/0	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 1-26**. 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 GigabitEthernet0/1/0 10.1.1.2
[*LSRA] ip route-static 3.3.3.3 32 GigabitEthernet0/2/0 10.2.1.2
[*LSRA] ip route-static 10.3.1.1 30 GigabitEthernet0/1/0
[*LSRA] ip route-static 10.4.1.1 30 GigabitEthernet0/2/0
[*LSRA] ip route-static 4.4.4.4 32 GigabitEthernet0/1/0 10.1.1.2 preference 40
[*LSRA] ip route-static 4.4.4.4 32 GigabitEthernet0/2/0 10.2.1.2 preference 60
[*LSRA] commit
```

Configure LSRB.

```
[~LSRB] ip route-static 1.1.1.1 32 GigabitEthernet0/1/0 10.1.1.1
[*LSRB] ip route-static 4.4.4.4 32 GigabitEthernet0/2/0 10.3.1.2
[*LSRB] commit
```

Configure LSRC.

```
[~LSRC] ip route-static 1.1.1.1 32 GigabitEthernet0/1/0 10.2.1.1
[*LSRC] ip route-static 4.4.4.4 32 GigabitEthernet0/2/0 10.4.1.2
[*LSRC] commit
```

Configure LSRD.

```
[~LSRD] ip route-static 2.2.2.2 32 GigabitEthernet0/1/0 10.3.1.1
[*LSRD] ip route-static 3.3.3.3 32 GigabitEthernet0/2/0 10.4.1.1
[*LSRD] ip route-static 10.1.1.2 30 GigabitEthernet0/1/0
[*LSRD] ip route-static 10.2.1.2 30 GigabitEthernet0/2/0
[*LSRD] ip route-static 1.1.1.1 32 GigabitEthernet0/1/0 10.3.1.1 preference 40
[*LSRD] ip route-static 1.1.1.1 32 GigabitEthernet0/2/0 10.4.1.1 preference 60
[*LSRD] commit
```

 Copyright © Huawei Technologies Co., Ltd.

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        GigabitEthernet0/1/0
        3.3.3.3/32  Static  60   0           D   10.2.1.1        GigabitEthernet0/2/0
        4.4.4.4/32  Static  40   0           D   10.1.1.1        GigabitEthernet0/1/0
       10.3.1.0/30  Static  60   0           D   10.1.1.1        GigabitEthernet0/1/0
       10.4.1.0/30  Static  60   0           D   10.2.1.1        GigabitEthernet0/2/0

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 GigabitEthernet0/1/0
[~LSRA-GigabitEthernet0/1/0] mpls
[*LSRA-GigabitEthernet0/1/0] mpls ldp
[*LSRA-GigabitEthernet0/1/0] commit
[~LSRA-GigabitEthernet0/1/0] quit
[~LSRA] interface GigabitEthernet0/2/0
[~LSRA-GigabitEthernet0/2/0] mpls
[*LSRA-GigabitEthernet0/2/0] mpls ldp
[*LSRA-GigabitEthernet0/2/0] commit
[~LSRA-GigabitEthernet0/2/0] 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 GigabitEthernet0/1/0 10.1.1.2 preference 40 ldp-sync
[*LSRA] interface GigabitEthernet0/1/0
[*LSRA-GigabitEthernet0/1/0] static-route timer ldp-sync hold-down 20
[*LSRA-GigabitEthernet0/1/0] commit
[~LSRA-GigabitEthernet0/1/0] quit
```

Configure LSRD.

```
[~LSRD] ip route-static 1.1.1.1 32 GigabitEthernet0/1/0 10.3.1.1 preference 40 ldp-sync
[*LSRD] interface GigabitEthernet0/1/0
[*LSRD-GigabitEthernet0/1/0] static-route timer ldp-sync hold-down 20
[*LSRD-GigabitEthernet0/1/0] commit
[~LSRD-GigabitEthernet0/1/0] 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 GigabitEthernet0/1/0
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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.252
 static-route timer ldp-sync hold-down 20
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0 10.1.1.2
 ip route-static 3.3.3.3 255.255.255.255 GigabitEthernet0/2/0 10.2.1.2
 ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet0/1/0 10.1.1.2 preference 40 ldp-sync
 ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet0/2/0 10.2.1.2
 ip route-static 10.3.1.0 255.255.255.252 GigabitEthernet0/1/0
 ip route-static 10.4.1.0 255.255.255.252 GigabitEthernet0/2/0
#
return
```

- LSRB configuration file

```
#
sysname LSRB
#
mpls lsr-id 2.2.2.2
#
mpls
#
mpls ldp
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.252
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0 10.1.1.1
 ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet0/2/0 10.3.1.2
#
return
```

- LSRC configuration file

```
#
sysname LSRC
#
mpls lsr-id 3.3.3.3
#
mpls
#
mpls ldp
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.2.1.2 255.255.255.252
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0 10.2.1.1
 ip route-static 4.4.4.4 255.255.255.255 GigabitEthernet0/2/0 10.4.1.2
#
return
```

- LSRD configuration file

```
#
sysname LSRD
#
mpls lsr-id 4.4.4.4
#
mpls
#
mpls ldp
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.3.1.2 255.255.255.252
 static-route timer ldp-sync hold-down 20
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0 10.3.1.1 preference 40 ldp-sync
 ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet0/2/0 10.4.1.1
 ip route-static 2.2.2.2 255.255.255.255 GigabitEthernet0/1/0 10.3.1.1
 ip route-static 3.3.3.3 255.255.255.255 GigabitEthernet0/2/0 10.4.1.1
 ip route-static 10.1.1.0 255.255.255.252 GigabitEthernet0/1/0
 ip route-static 10.2.1.0 255.255.255.252 GigabitEthernet0/2/0
#
return
```

1.1.4 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.

1.1.4.1 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.

1.1.4.2 Configuration Precautions for IPv6 Static Route

Feature Requirements

None

1.1.4.3 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.

1.1.4.3.1 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

1.1.4.3.2 (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

1.1.4.3.3 (Optional) Enabling a Device to Recurse IPv6 Static Routes to ARP 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 ARP 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 ARP Vlink direct routes. This prevents blackhole routes from being generated, thereby preventing traffic loss.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ipv6 route recursive-lookup arp vlink-direct-route protocol static**

The device is enabled to recurse IPv6 static routes to ARP Vlink direct routes.

This command is mainly used in scenarios where a Layer 2 network accesses a Layer 3 network. To allow IPv6 static routes to recurse only to ARP Vlink direct routes, you also need to run the **ipv6 route-static** command with the **recursive-lookup host-route** parameter specified.

Step 3 Run **commit**

The configuration is committed.

----End

1.1.4.3.4 (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

1.1.4.3.5 (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

1.1.4.3.6 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

1.1.4.4 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.4.5 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.

1.1.4.5.1 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**

Global BFD is enabled on the node.

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* }*

Step 5 Run **ipv6 route-static bfd** [*interface-type interface-number* | **vpn-instance** *vpn-destination-name*] *nexthop-address* **local-address** *address* [**detect-multiplier** *multiplier*] [**min-rx-interval** *min-rx-interval*] [**min-tx-interval** *min-tx-interval*] *

The BFD parameters of an IPv6 static route are set.

NOTE

If *interface-type interface-number* is not set, **local-address** *ipv6-address* must be specified.

If none of min-rx-interval, min-tx-interval, or detect-multiplier is specified, the default values of the BFD parameters are used globally.

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

Checking the Configurations

Run the following commands to check the previous 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.

1.1.4.5.2 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.

1. Run **system-view**

   The system view is displayed.
2. Run **bfd**

   BFD is globally enabled.
3. Run **quit**

   The system view is displayed.
4. 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.
5. Run **discriminator local** *discr-value*

   The local discriminator of a static BFD session is set.
6. 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 NE40E-M2 series Universal Service Router Configuration Guide - Network Reliability*.

7. Run **commit**

   The configuration is committed.

Step 2 Associate an IPv6 static route with a BFD session.

1. 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.

1.1.4.5.3 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.
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 NE40E-M2 series Universal Service 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* | *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* | *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.

1.1.4.6 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 1-27**, 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 1-27 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.

1.1.4.7 Configuration Examples for IPv6 Static Routes

This section provides configuration examples of IPv6 static routes.

1.1.4.7.1 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 1-28**, 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 1-28 Networking for configuring IPv6 static routes

NOTE

Interfaces 1 through 3 in this example represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 0/1/0 as the outbound interface on Device A
- Static route to 2001:db8:1:: 64 with GE 0/1/0 as the outbound interface on Device B
- Static route to 2001:db8:3:: 64 with GE 0/2/0 as the outbound interface on Device B
- Default route with GE 0/1/0 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 0/1/0 2001:db8:4::2
[*DeviceA] commit
```

Configure two IPv6 static routes on Device B.

```
[~DeviceB] ipv6 route-static 2001:db8:1:: 64 gigabitethernet 0/1/0 2001:db8:4::1
[*DeviceB] ipv6 route-static 2001:db8:3:: 64 gigabitethernet 0/2/0 2001:db8:5::1
[*DeviceB] commit
```

Configure an IPv6 default route on Device C.

```
[~DeviceC] ipv6 route-static :: 0 gigabitethernet 0/1/0 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    : GigabitEthernet0/1/0        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  : ::FFFF: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    : GigabitEthernet0/2/0        Flags        : D

Destination  : 2001:DB8:1::1               PrefixLength : 128
NextHop      : 2001:DB8::1                 Preference   : 0
```

```
Cost         : 0                          Protocol     : Direct
RelayNextHop : ::                             TunnelID     : 0x0
Interface    : GigabitEthernet0/2/0              Flags        : D

Destination  : 2001:DB8:4::                     PrefixLength : 64
NextHop      : 2001:DB8:4::1                     Preference   : 0
Cost         : 0                          Protocol     : Direct
RelayNextHop : ::                             TunnelID     : 0x0
Interface    : GigabitEthernet0/1/0              Flags        : D

Destination  : 2001:DB8:4::1                    PrefixLength : 128
NextHop      : 2001:DB8::1                      Preference   : 0
Cost         : 0                          Protocol     : Direct
RelayNextHop : ::                             TunnelID     : 0x0
Interface    : GigabitEthernet0/1/0              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 GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:4::1/64
#
```

```
ipv6 route-static :: 0 GigabitEthernet 0/1/0 2001:db8:4::2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:4::2/64
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:5::2/64
#
interface GigabitEthernet0/3/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/64
#
ipv6 route-static 2001:db8:1:: 64 GigabitEthernet0/1/0 2001:db8:4::1
ipv6 route-static 2001:db8:3:: 64 GigabitEthernet0/2/0 2001:db8:5::1
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:5::1/64
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::1/64
#
ipv6 route-static :: 0 GigabitEthernet0/1/0 2001:db8:5::2
#
return
```

1.1.4.7.2 Example for Configuring IPv6 Floating Static Routes

IPv6 Floating static routes can be used for the static route backup.

Networking Requirements

Figure Networking for configuring IPv6 floating static routes 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 1-29 Networking for configuring IPv6 floating static routes

NOTE

Interfaces 1 through 5 in this example represent GE 0/1/1, GE 0/1/2, GE 0/1/3, GE 0/2/3, and GE 0/3/3, 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 IPv6 static routes to 2001:db8:3::/64 and 2001:db8:4::/64.
4. On Device A, configure two IPv6 static routes to 2001:db8:5::/64 with different priorities.
5. 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:db8:3:: 64 2001:db8:1::2
[*DeviceA] ipv6 route-static 2001:db8:4:: 64 2001:db8:2::2
[*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 : 3

Static routing table status : <Active>
Summary Count : 3

Destination  : 2001:db8:3::                    PrefixLength : 64
NextHop      : 2001:db8:1::2                    Preference   : 60
Cost         : 0                            Protocol     : Static
RelayNextHop : 2001:db8:1::2                      TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1               Flags        : RD

Destination  : 2001:db8:4::                    PrefixLength : 64
NextHop      : 2001:db8:2::2                    Preference   : 60
Cost         : 0                            Protocol     : Static
RelayNextHop : 2001:db8:2::2                      TunnelID     : 0x0
Interface    : GigabitEthernet0/1/2               Flags        : RD

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    : GigabitEthernet0/1/1               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 0/1/1 of Device A to simulate a link fault.

```
[~DeviceA] interface GigabitEthernet 0/1/1
[~DeviceA-GigabitEthernet0/1/1] shutdown
[*DeviceA-GigabitEthernet0/1/1] commit
[~DeviceA-GigabitEthernet0/1/1] 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 : 2

Static routing table status : <Active>
Summary Count : 2

Destination  : 2001:db8:4::                    PrefixLength : 64
NextHop      : 2001:db8:2::2                    Preference   : 60
Cost         : 0                          Protocol     : Static
RelayNextHop : 2001:db8:2::2                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/2             Flags        : RD

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    : GigabitEthernet0/1/2             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 GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/64
#
ipv6 route-static 2001:db8:3:: 64 2001:db8:1::2
ipv6 route-static 2001:db8:4:: 64 2001:db8:2::2
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 GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
#
interface GigabitEthernet0/1/3
```

```
 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 GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
#
interface GigabitEthernet0/2/3
 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 GigabitEthernet0/1/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::2/64
#
interface GigabitEthernet0/2/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:4::2/64
#
interface GigabitEthernet0/3/3
 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
```

1.1.4.7.3 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 1-30**, 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 1-30 Networking for configuring dynamic BFD for IPv6 static routes

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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    : GigabitEthernet0/2/0        Flags        : D

Destination  : 2001:db8:7::1               PrefixLength : 128
NextHop      : ::1                         Preference   : 0
Cost         : 0                           Protocol     : Direct
RelayNextHop : ::                          TunnelID     : 0x0
Interface    : GigabitEthernet0/2/0        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    : GigabitEthernet0/1/0        Flags        : RD

Destination  : 2001:db8:200::              PrefixLength : 64
NextHop      : 2001:db8:200::1             Preference   : 0
Cost         : 0                           Protocol     : Direct
RelayNextHop : ::                          TunnelID     : 0x0
Interface    : GigabitEthernet0/1/0        Flags        : D

Destination  : 2001:db8:200::1             PrefixLength : 128
NextHop      : ::1                         Preference   : 0
Cost         : 0                           Protocol     : Direct
RelayNextHop : ::                          TunnelID     : 0x0
Interface    : GigabitEthernet0/1/0        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    : -
-------------------------------------------------------------------------------

   Total UP/DOWN Session Number : 1/0
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
bfd
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:200::1/64
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:200::2/64
#
interface GigabitEthernet0/2/0
 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
```

1.1.4.7.4 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 1-31**, 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 1-31 Configuring static BFD for IPv6 static routes

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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    : GigabitEthernet0/1/0           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    : GigabitEthernet0/1/0          Flags        : D

 Destination  : 2001:db8:1::1                PrefixLength : 128
 NextHop      : 2001:db8:3::1                Preference   : 0
 Cost         : 0                         Protocol     : Direct
 RelayNextHop : ::                           TunnelID     : 0x0
 Interface    : GigabitEthernet0/1/0          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 0/1/0 of Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet0/1/0
[~DeviceB-GigabitEthernet0/1/0] 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
#
interface GigabitEthernet0/2/0
 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
```

1.1.4.7.5 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 1-32**, 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).

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-32 NQA for IPv6 static routes

Device Name	Interface	IP Address
DeviceA	GE 0/1/0	2001:db8:1::1/64
	GE 0/2/3	2001:db8:2::1/64
DeviceB	GE 0/1/0	2001:db8:1::2/64
	GE 0/1/1	2001:db8:3::1/64
DeviceC	GE 0/1/0	2001:db8:6::1/64
	GE 0/2/3	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 1-5**)

Table 1-5 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)

 Copyright © Huawei Technologies Co., Ltd.

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 0/1/1 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 0/1/0 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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
[~DeviceA] interface GigabitEthernet 0/2/3
[*DeviceA-GigabitEthernet0/2/3] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet0/2/3] commit
[~DeviceA-GigabitEthernet0/2/3] quit
```

Configure OSPFv3 on Device B.

```
[~DeviceB] interface GigabitEthernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] quit
```

Configure OSPFv3 on Device C.

```
[~DeviceC] interface GigabitEthernet 0/2/3
[~DeviceC-GigabitEthernet0/2/3] ospfv3 1 area 0.0.0.0
[*DeviceC-GigabitEthernet0/2/3] commit
[~DeviceC-GigabitEthernet0/2/3] 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 GigabitEthernet0/1/1 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    : GigabitEthernet0/1/1            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    : GigabitEthernet0/1/0            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 0/1/0 and the cost is 10. Traffic is preferentially transmitted along the link Device B -> Switch A.

Shut down GE 0/1/1 on Device B to simulate a link fault.

```
[~DeviceB] interface GigabitEthernet 0/1/1
[*DeviceB-GigabitEthernet0/1/1] shutdown
[*DeviceB-GigabitEthernet0/1/1] commit
[~DeviceB-GigabitEthernet0/1/1] 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          Preference   : 150
Cost         : 20                              Protocol     : OSPFv3ASE
RelayNextHop : ::                              TunnelID     : 0x0
Interface    : GigabitEthernet0/1/0            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        Preference   : 150
Cost         : 20                              Protocol     : OSPFv3ASE
RelayNextHop : ::                              TunnelID     : 0x0
Interface    : GigabitEthernet0/2/3            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 0/2/3, 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 0.0.0.0
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/2/3
 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 0.0.0.0
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::1/64
#
interface GigabitEthernet0/1/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:5::1/64
#
ipv6 route-static 2001:db8:7:: 64 GigabitEthernet0/1/1 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 0.0.0.0
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:6::1/64
#
interface GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:4::1/64
#
interface GigabitEthernet0/2/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
 ospfv3 1 area 0.0.0.0
#
ipv6 route-static 2001:db8:7:: 64 GigabitEthernet0/1/0 2001:db8:6::2
#
return
```

1.1.4.7.6 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 1-33**, 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 1-33 Networking for configuring FRR for IPv6 static routes on the public network

NOTE

Interfaces 1 through 5 in this example represent GE 0/1/1, GE 0/1/2, GE 0/1/3, GE 0/2/3, and GE 0/3/3, 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. Static BFD for IPv6 static routes is used as an example because it is more common than dynamic BFD for IPv6 static routes 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 priorities 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 GigabitEthernet0/1/2 2001:db8:1::1
[*DeviceA] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/2/3 2001:db8:3::2
[*DeviceA] commit
```

On Device B, configure static routes.

```
[~DeviceB] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet0/1/1 2001:db8:2::1
[*DeviceB] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/3/3 2001:db8:4::2
[*DeviceB] commit
```

On Device C, configure IPv6 static routes.

```
[~DeviceC] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet0/2/3 2001:db8:3::1
[*DeviceC] ipv6 route-static 2001:db8:5:: 64 GigabitEthernet0/3/3 2001:db8:4::1
[*DeviceC] ipv6 route-static 2001:db8:1:: 64 GigabitEthernet0/2/3 2001:db8:3::1
[*DeviceC] ipv6 route-static 2001:db8:2:: 64 GigabitEthernet0/3/3 2001:db8:4::1
[*DeviceC] commit
```

On Device D, configure IPv6 static routes.

```
[~DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/1/2 2001:db8:1::2
[*DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/1/1 2001:db8:2::2
[*DeviceD] ipv6 route-static 2001:db8:3:: 64 GigabitEthernet0/1/2 2001:db8:1::2
[*DeviceD] ipv6 route-static 2001:db8:4:: 64 GigabitEthernet0/1/1 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 IPv6static 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    : GigabitEthernet0/1/2         Flags        : D

Destination  : 2001:db8:1::1                PrefixLength : 128
NextHop      : ::1                          Preference   : 0
Cost         : 0                            Protocol     : Direct
RelayNextHop : ::                           TunnelID     : 0x0
Interface    : GigabitEthernet0/1/2         Flags        : D

Destination  : 2001:DB8:2::                 PrefixLength : 64
```

```
NextHop      : 2001:DB8:2::1                    Preference   : 0
Cost         : 0                              Protocol     : Direct
RelayNextHop : ::                                 TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1                     Flags        : D

Destination  : 2001:DB8:2::1                    PrefixLength : 128
NextHop      : ::1                              Preference   : 0
Cost         : 0                              Protocol     : Direct
RelayNextHop : ::                                 TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1                     Flags        : D

Destination  : 2001:DB8:3::                     PrefixLength : 64
NextHop      : 2001:DB8:1::2                    Preference   : 60
Cost         : 0                              Protocol     : Static
RelayNextHop : ::                                 TunnelID     : 0x0
Interface    : GigabitEthernet0/1/2                     Flags        : D

Destination  : 2001:DB8:4::                     PrefixLength : 64
NextHop      : 2001:DB8:2::2                    Preference   : 60
Cost         : 0                              Protocol     : Static
RelayNextHop : ::                                 TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1                     Flags        : D

Destination  : 2001:db8:5::                     PrefixLength : 64
NextHop      : 2001:db8:5::1                    Preference   : 0
Cost         : 0                              Protocol     : Direct
RelayNextHop : ::                                 TunnelID     : 0x0
Interface    : GigabitEthernet0/1/3                     Flags        : D

Destination  : 2001:db8:5::1                    PrefixLength : 128
NextHop      : ::1                              Preference   : 0
Cost         : 0                              Protocol     : Direct
RelayNextHop : ::                                 TunnelID     : 0x0
Interface    : GigabitEthernet0/1/3                     Flags        : D

Destination  : 2001:db8:6::                     PrefixLength : 64 NextHop      :
2001:db8:2::2                    Preference   : 60 Cost        : 0                              Protocol     :
Static RelayNextHop : ::                                 TunnelID     : 0x0 Interface    :
GigabitEthernet0/1/1                 Flags        : D

Destination  : 2001:db8:6::                     PrefixLength : 64 NextHop      :
2001:db8:1::2                    Preference   : 60 Cost        : 0                              Protocol     :
Static RelayNextHop : ::                                 TunnelID     : 0x0 Interface    :
GigabitEthernet0/1/2                 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 priorities of static routes on Device D.

```
<DeviceD> system-view
[~DeviceD] ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/1/2 2001:db8:1::2 preference 40
[*DeviceD] commit
[*DeviceD] quit
```

Check the IP routing table of Device D. The following command output shows that the priorities of IPv6 static routes have 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    : GigabitEthernet0/1/2  Flags        : D

Destination  : 2001:db8:1::1         PrefixLength : 128
NextHop      : ::1                   Preference   : 0
Cost         : 0                     Protocol     : Direct
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/2  Flags        : D

Destination  : 2001:DB8:2::          PrefixLength : 64
NextHop      : 2001:DB8:2::1         Preference   : 0
Cost         : 0                     Protocol     : Direct
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1  Flags        : D

Destination  : 2001:DB8:2::1         PrefixLength : 128
NextHop      : ::1                   Preference   : 0
Cost         : 0                     Protocol     : Direct
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1  Flags        : D

Destination  : 2001:DB8:3::          PrefixLength : 64
NextHop      : 2001:DB8:1::2         Preference   : 60
Cost         : 0                     Protocol     : Static
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/2  Flags        : D

Destination  : 2001:DB8:4::          PrefixLength : 64
NextHop      : 2001:DB8:2::2         Preference   : 60
Cost         : 0                     Protocol     : Static
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1  Flags        : D

Destination  : 2001:db8:5::          PrefixLength : 64
NextHop      : 2001:db8:5::1         Preference   : 0
Cost         : 0                     Protocol     : Direct
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/3  Flags        : D

Destination  : 2001:db8:5::1         PrefixLength : 128
NextHop      : ::1                   Preference   : 0
Cost         : 0                     Protocol     : Direct
RelayNextHop : ::                    TunnelID     : 0x0
Interface    : GigabitEthernet0/1/3  Flags        : D

Destination  : 2001:db8:6::          PrefixLength : 64 NextHop      :
2001:db8:1::2                Preference   : 40 Cost         : 0                    Protocol     :
Static RelayNextHop : ::                    TunnelID     : 0x0 Interface    :
GigabitEthernet0/1/2                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    : GigabitEthernet0/1/2        Flags        : D
BkNextHop    : 2001:db8:2::2               BkInterface  : GigabitEthernet0/1/1
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 GigabitEthernet0/1/2 2001:db8:1::2 preference 40
track bfd-session aa
```

Step 6 Simulate a fault on Link A.

```
[~DeviceD] interface GigabitEthernet0/1/2
[~DeviceD-GigabitEthernet0/1/2] shutdown
```

```
[~DeviceD-GigabitEthernet0/1/2] commit
[~DeviceD-GigabitEthernet0/1/2] quit
[~DeviceD] quit
```

Check the routes to 172.17.1.0/24 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
RelayNextHop : ::                              TunnelID     : 0x0
Interface    : GigabitEthernet0/1/1            Flags        : D
```

----End

Configuration Files

- Device D configuration file

```
#
sysname DeviceD
#
bfd
#
interface GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/64
#
interface GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet0/1/3
 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 GigabitEthernet0/1/2 2001:db8:1::2 preference 40 track bfd-session aa
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/1/1 2001:db8:2::2
ipv6 route-static 2001:db8:3:: 64 GigabitEthernet0/1/2 2001:db8:1::2
ipv6 route-static 2001:db8:4:: 64 GigabitEthernet0/1/1 2001:db8:2::2
#
return
```

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
#
```

```
interface GigabitEthernet0/2/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::1/64
#
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet0/1/2 2001:db8:1::1
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/2/3 2001:db8:3::2
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
#
interface GigabitEthernet0/3/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:4::1/64
#
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet0/1/1 2001:db8:2::1
ipv6 route-static 2001:db8:6:: 64 GigabitEthernet0/3/3 2001:db8:4::2
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
bfd
#
interface GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:6::1/64
#
interface GigabitEthernet0/2/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::2/64
#
interface GigabitEthernet0/3/3
 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 GigabitEthernet0/2/3 2001:db8:3::1
ipv6 route-static 2001:db8:5:: 64 GigabitEthernet0/3/3 2001:db8:4::1
ipv6 route-static 2001:db8:1:: 64 GigabitEthernet0/2/3 2001:db8:3::1
ipv6 route-static 2001:db8:2:: 64 GigabitEthernet0/3/3 2001:db8:4::1
#
return
```

1.1.5 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.

1.1.5.1 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 1-6** describes differences among the three typical IGPs.

Table 1-6 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

 Copyright © Huawei Technologies Co., Ltd.

- Multicast: OSPF uses multicast addresses to send packets on certain types of links, which minimizes the impact on other devices.

1.1.5.2 Configuration Precautions for OSPF

Feature Requirements

Table 1-7 Feature requirements

Feature Requirements	Series	Models
OSPF checks whether a neighbor in the full state exists in the backbone area before advertising a default route to the stub area.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
1. The mapping SID priority is not supported. All mapping SIDs are processed based on the default priority. 2. On the TiLfa protection path, all devices from the source node to the Q node must support segment routing, and an LDP path must exist between the Q node and the destination node. 3. SR routes mapped to mapping SIDs do not support anycast FRR. Anycast protection is not implemented between SR routes mapped to mapping SIDs or between SR routes mapped to mapping SIDs and common SR routes. 4. SR routes mapped to mapping SIDs do not support remote microloop avoidance. 5. In inter-process and inter-protocol scenarios, when an ASBR functions as an interworking stitching node, MappingSid mapping needs to be configured on the imported process. Plan the configuration properly.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In SR and LDP interworking scenarios, PHP is not supported. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In SR and LDP interworking scenarios, only the mapping SID with the 32-bit mask IP prefix is supported. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In an SR and LDP interworking scenario, FRR does not take effect in some scenarios, for exmaple: In the SR to LDP direction, the Q node and all NEs on the path before the Q node are required to support SR. In the LDP to SR interworking FRR scenario, the PQ and all NEs on the path before the PQ are required support LDP. Properly plan configurations.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In an OSPF loop detection scenario, when the maximum cost is advertised due to a loop, the apply cost command does not take effect.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
OSPF broadcast networks support association between link BFD and cost values. 1) When only one neighbor relationship is established and BFD goes Down, the neighbor relationship is not deleted, and the OSPF LSDB component is notified that the neighbor relationship enters the BFD-associated cost state. 2) This function is not supported when there are multiple neighbors. When the number of neighbors changes from one to multiple, the OSPF LSDB component is notified of the deletion of the neighbor in the BFD Down state. It is recommended that only one neighbor relationship be established on an OSPF broadcast network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
SBFD is not supported in SR and LDP interworking scenarios because LDP does not support SBFD. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In an OSPF loop detection scenario, after a loop is removed, the loop cannot be automatically removed and the alarm cannot be automatically cleared. You need to manually configure a routing policy or route tag and run the clear route loop-detect ospf alarm-state command to remove the route from the loop state and clear the alarm.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In OSPF loop detection scenarios, aggregated routes and NSSA routes do not support loop detection.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
OSPF loop detection does not support router ID conflicts, including intra-AS and inter-AS router ID conflicts. A router ID conflict may trigger incorrect detection.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In OSPF loop detection scenarios, a single process supports loop detection for a maximum of 100,000 routes. If the number of routes exceeds 100,000, loop detection cannot be performed.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
After the device enters the loop breaking state, it cannot automatically exit the routing loop state and the alarm cannot be automatically cleared. In this case, manual intervention is required. For example, after a routing policy or route tag is correctly configured, run the clear route loop-detect ospf alarm-state command to exit the routing loop state and clear the alarm.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In OSPF loop detection scenarios, the default route advertised using the default-route-advertise command supports only loop detection and does not support self-healing.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.5.3 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.

1.1.5.3.1 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 NE40E 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 NE40E 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 **1.1.5.4.1 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

1.1.5.3.2 Configuring an Interface to Fill in a DD Packet with the Interface 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

1.1.5.3.3 (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

1.1.5.3.4 (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

1.1.5.3.5 (Optional) Setting the OSPF Priority

When multiple routing protocols are used to select routes, you can set the OSPF priority to maneuver route selection.

Context

The routing protocols may share and select the routing information because the Router may run multiple dynamic routing protocols at the same time. 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 process view is displayed.

Step 3 Run **preference** [**ase** | **inter** | **intra**] { *preference* | { **route-policy** *route-policy-name* | **route-filter** *route-filter-name* } } *

The OSPF priority is set.

- **ase**: sets the priority of the AS-External route.
- **inter**: sets the priority of the Inter-area route.
- **intra**: sets the priority of the Intra-area route.
- *preference*: sets the priority for OSPF. The smaller the value, the higher the priority.
- *route-policy-name*: sets the priority for specified routes in the routing policy.
- *route-filter-name*

  : sets the priority for specified routes in the route-filter policy.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.3.6 (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

1.1.5.3.7 (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 process view is displayed.

Step 3 Run **retransmission-limit** [*max-number*]

The maximum number of OSPF packet retransmission attempts is set.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.3.8 (Optional) Setting the Interval at Which LSAs Are Retransmitted Between Adjacent Routers

Control packet retransmission and improve the convergence rate by setting an interval at which an LSA packet is retransmitted to the adjacent Router.

Context

After sending an LSA packet to the adjacent Router, the Router 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 2^2

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 on the Router. 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 Routers.

The default value is recommended.

Step 4 Run **commit**

 Copyright © Huawei Technologies Co., Ltd.

The configuration is committed.

----End

1.1.5.3.9 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

1.1.5.4 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 1-8**.

NOTE

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 1-8 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.	- If the link layer protocol is ATM, OSPF regards the network as an NBMA network by default.
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, HDLC, or 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 Routers 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.

 Copyright © Huawei Technologies Co., Ltd.

As shown in **Table 1-8**, 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**.

1.1.5.4.1 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**, the network type of serial interfaces is **p2p**, the network type of Frame-relay interfaces is **nbma**.

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** [**peer-ip-ignore**] }

A network type is configured for the OSPF interface.

When 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 an interface is broadcast but some Routers on the broadcast network do not support multicast addresses, you can change the network type of the interface to NBMA.
- If the network type of an interface is NBMA and the network is fully meshed (any two Routers are directly reachable), you can change the interface type to broadcast, without the need to configure neighbor Routers.
- If the network type of an interface is NBMA but the network is not fully meshed, you need to change the network type of the interface to P2MP. In this manner, two Routers that are not directly reachable can exchange routing

information through one Router that is directly reachable to the two Routers. After the network type of the interface is changed to P2MP, there is no need to configure neighbor Routers.

- If only two Routers run OSPF on the same network segment, changing the interface network type to P2MP is recommended.

peer-ip-ignore is used to disable network segment check during OSPF neighbor relationship establishment on an interface for which the network type is changed from broadcast to P2P and address unnumbering is not configured. For network segment check, the local interface performs an AND operation between its mask and the local interface address and between this mask and the peer interface address. If the results are the same, an OSPF neighbor relationship can be established. If the results are different, no OSPF neighbor relationship can be established.

NOTE

OSPF does not support the configuration of a null interface.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.4.2 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, run specific commands to forcibly change the network type to NBMA. 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*

Sets the priority of an interface that runs for the DR or BDR.

The priority of an interface determines whether the interface is qualified to be a DR or BDR. The interface with the highest priority is elected as the DR or BDR. 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*

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 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 from the interface view.
2. Run **ospf** [*process-id*]

   The OSPF process view is displayed.
3. Run **peer** *ip-address* [**dr-priority** *priority*]

   A neighboring Router is configured.
4. Run **commit**

   The configuration is committed.

Step 6 Run **commit**

The configuration is committed.

----End

1.1.5.4.3 Configuring P2MP Network Attributes

To implement OSPF functions, configure P2MP network attributes.

Procedure

Step 1 Disable OSPF from checking 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 Router 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

1.1.5.4.4 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

1.1.5.5 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**.

1.1.5.5.1 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.
  a. Run **system-view**

     The system view is displayed.
  b. Run **interface** *interface-type interface-number*

     The interface view is displayed.
  c. Run **ospf cost** *value*

     A cost is configured for the OSPF interface.
  d. 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.

----End

1.1.5.5.2 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 1-34**, 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 1-34 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 process 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 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

1.1.5.5.3 Setting the convergence priority for OSPF routes

You can adjust and optimize route selection by setting the convergence priority for OSPF routes.

Context

To set the convergence priority of OSPF routes based on a specified IP prefix list 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. This command enables OSPF to calculate 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

1.1.5.5.4 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 process view is displayed.

Step 3 Run **stub-router** [[**on-startup** [*interval*]] | [**include-stub**] | [**external-lsa** [*externallsa-metric*]] | [**summary-lsa** [*summarylsa-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 is set for OSPF links.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.5.5.5 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 Device A and Device B, as shown in **Figure 1-35**. To configure the route with the outbound interface of Interface 2 to be the optimal route, suppress Interface 1 and Interface 3 from receiving and sending OSPF packets.

Figure 1-35 Networking diagram of suppressing an interface from receiving and sending OSPF packets

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospf** [*process-id*]

The OSPF process 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

1.1.5.5.6 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

1.1.5.5.7 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

1.1.5.5.8 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

1.1.5.6 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**.

1.1.5.6.1 Configuring OSPF to Import External Routes

Importing the routes discovered by other routing protocols can enrich OSPF routing information.

Context

To access a Router running a non-OSPF protocol, an OSPF-capable Router needs to import routes of the non-OSPF 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 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 **import-route** { **bgp** [**permit-ibgp**] | **direct** | **rip** [*process-id-rip*] | **static** | **unr** | **isis** [*process-id-isis*] | **ospf** [*process-id-ospf*] } [**cost** *cost* | **tag** *tag* | **type** *type* | **route-policy** *route-policy-name* | **route-filter** *route-filter-name*] *

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

1.1.5.6.2 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.

The default route is 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 1-9**.

Table 1-9 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 process 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.

NOTE

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 Configuring an OSPF NSSA.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.6.3 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

- Configuring 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.

Perform the following steps on the OSPF ASBR:

a. Run **system-view**

The system view is displayed.

b. Run **ospf** [*process-id*]

The OSPF process view is displayed.

c. Run **asbr-summary** *ip-address mask* [[**not-advertise** | **generate-null0-route**] | **tag** *tag-value* | **cost** *cost-value* | **distribute-delay** *interval*] *

ASBR route summarization is configured.

d. Run **commit**

The configuration is committed.

NOTE

After route summarization is configured, the routing table on the local OSPF router remains the same. The routing table on another OSPF router, however, contains only one summarized route, no specific route. This summarized route is not removed until all specific routes are interrupted.

----End

1.1.5.6.4 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:

- ○ 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* } **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 XPL 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

1.1.5.6.5 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

1.1.5.6.6 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 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 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

1.1.5.6.7 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 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* } **export** [**direct** | **static** | **unr** | **bgp** | { **rip** | **isis** | **ospf** } [*process-id*]]

   An export policy that is based on the basic ACL is configured for OSPF to filter 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*]]

  An export policy that is based on the IP prefix list is configured for OSPF to filter 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*]]

  An export policy that is based on the route-policy is configured for OSPF to filter imported routes to be advertised.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.6.8 (Optional) Configuring OSPF to Discard Specified LSAs

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

1.1.5.6.9 Configuring the Maximum Number of External Routes Supported by the OSPF LSDB

You can set the maximum number of external routes in the LDSB 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*

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

1.1.5.6.10 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

1.1.5.7 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**.

1.1.5.7.1 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.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.7.2 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

1.1.5.7.3 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

1.1.5.7.4 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 process view is displayed.

Step 3 Run **sham-hello enable**

OSPF sham hello is enabled.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.5.7.5 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

1.1.5.7.6 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 ≥ 2) time equals *hold-interval* x $2^{(n-2)}$.
3. When the interval specified by *hold-interval* x $2^{(n-2)}$ 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 the OSPF LSAs to be sent flap is configured.

If no flapping occurs among the OSPF LSAs to be sent, the configuration of the **lsa-originate-interval** command prevents the device from frequently sending LSAs. If the OSPF LSAs to be sent flap, the configuration of the **lsa-originate-interval suppress-flapping** command reduces 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

1.1.5.7.7 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 ≥ 2) time equals *hold-interval* x $2^{(n-1)}$.
3. When the interval specified by *hold-interval* x $2^{(n-1)}$ 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. The larger value of the two intervals is used as the suppression period.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.5.7.8 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.

 Copyright © Huawei Technologies Co., Ltd.

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 ≥ 2) time equals *hold-interval* x $2^{(n-2)}$.
3. After the interval specified by *hold-interval* x $2^{(n-2)}$ 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

1.1.5.7.9 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 process view is displayed.
  c. Run **suppress-reachability**

The suppresses the advertisement of all interface addresses in an OSPF process is configured.

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

1.1.5.7.10 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 process 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

1.1.5.7.11 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) x 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

1.1.5.7.12 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

1.1.5.7.13 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

1.1.5.7.14 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

1.1.5.7.15 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

1.1.5.8 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 enabling delay advertisement, complete the following task:

- Configure the TWAMP Light controller to detect delay information.

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 advertisement enable**

Delay advertisement is enabled.

Step 5 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.

1.1.5.9 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.

NOTE

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.

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 OSPF neighbor relationship flapping suppression from one of the interfaces, run the **ospf suppress-flapping peer disable** command.

Step 3 Run **ospf suppress-flapping peer hold-down** *interval*

The Hold-down mode and its duration are configured.

OSPF neighbor relationship 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: 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.

  NOTE

  To change the maximum cost for OSPF links, run the **maximum-link-cost** *cost* command.

The Hold-down and Hold-max-cost modes can work together. When both modes take effect, 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** command.

 Copyright © Huawei Technologies Co., Ltd.

Step 4 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.

- 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 5 Run **quit**

The system view is displayed.

Step 6 Run **quit**

The user view is displayed.

Step 7 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.

Step 8 Run **commit**

The configuration is committed.

----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.

1.1.5.10 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.

1.1.5.11 Configuring Routing Loop Detection for Routes Imported into OSPF

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**

Exit the routing loop detection alarm state and the 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.

Step 3 Run **route loop-detect ospf enable**

The routing loop detection for routes imported to OSPF is enabled.

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

1.1.5.12 Configuring OSPF Flush LSA Source Tracing

OSPF flush LSA source tracing helps improve the efficiency in locating the devices that flushed OSPF LSAs.

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.

NOTE

The following steps are optional, choose them as required.

Pre-configuration Tasks

Before configuring OSPF flush LSA 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.

To disable OSPF flush LSA source tracing globally, run the **ospf flush-source-trace disable** command.

Step 2 Run **interface** *interface-type interface-number*

The interface view is displayed.

To disable OSPF flush LSA source tracing on an interface, run the **ospf flush-source-trace block** command.

Step 3 Run **quit**

The system view is displayed.

Step 4 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 5 Run **quit**

The user view is displayed.

Step 6 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.

Step 7 Run **commit**

The configuration is committed.

----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.

1.1.5.13 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

- Configure a dynamic OSPF hostname.
  a. Run **system-view**

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

     The OSPF process view is displayed.
  c. Run **opaque-capability enable**

     The Opaque LSA capability is enabled.
  d. Run **hostname** *host-name*

     An OSPF dynamic hostname is configured.

     NOTE

     If you specify *hostname* in **hostname** command, *hostname* is advertised as the dynamic hostname. If no *hostname* is specified in **hostname** command, the hostname specified in the **sysname** command is advertised as the dynamic hostname.
  e. Run **commit**

     The configuration is committed.

----End

Checking the Configurations

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**]

1.1.5.14 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 process 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.

NOTE

- 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.

1.1.5.15 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 **1.1.5.14 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 the received LSAs and translates them into Type 5 LSAs to advertise AS external routes to the other areas over the OSPF network.

NOTE

- A Type 7 LSA was introduced to support NSSAs and describe imported external routes.
- Type 7 LSAs can be used to carry default route to transmit 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 process 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 routing update in the area and disconnection from neighbors. The NSSA attributes can be reconfigured or re-canceled only after the routing update is complete.

The **nssa** command is applicable to the following scenarios:

- Default Type 7 LSAs are generated and advertised to an NSSA.
  a. OSPF neighbor relationships that are in the Full state and interfaces that are Up 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, ABRs generate default Type 7 LSAs and advertise them to the NSSA. If conditions 2 and 3 are met, ASBRs generate default Type 7 LSAs and advertise them to the NSSA.

- The parameter **backbone-peer-ignore** is used to prevent the ABR from checking the neighbor status when the ABR generates default Type 7 LSAs and advertises them to the NSSA. Specifically, the ABR generates default Type 7 LSAs and advertises them to the NSSA as long as an interface that is Up exist in the backbone area.
- An ASBR also functions as an ABR, and the **no-import-route** parameter is configured to prevent external routes imported using the **import-route** command from being advertised to the NSSA.
- The **no-summary** parameter is configured on an ABR to reduce the number of LSAs to be transmitted to the NSSA, which prevents the ABR from transmitting Type 3 LSAs to the NSSA.

  NOTE

  After the **nssa default-route-advertise backbone-peer-ignore no-summary** command is run, the ABR generates default Type 7 and Type 3 LSAs as long as an interface that is Up exist in the backbone area. The default Type 3 LSAs preferentially take effect.
- After the **set-n-bit** parameter is configured, the N-bit is set in the database description (DD) packets during the synchronization between the Router and neighboring Routers.
- The **suppress-forwarding-address** parameter sets the forwarding address (FA) of the Type 5 LSAs translated from Type 7 LSAs by the NSSA ABR to 0.0.0.0.
- If multiple ABRs are deployed in the NSSA, the system automatically selects an ABR (the Router with the largest router ID) as a translator to convert 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 specify two ABRs for load balancing, configure the **translator-always** parameter on the chosen ABRs to specify the ABRs as all-the-time translators. You can use this command to pre-configure a fixed translator to prevent LSA flooding caused by translator role changes.
- The **translator-interval** parameter is used to ensure uninterrupted services when translator roles change. 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 carried in 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.

Step 6 Run **commit**

The configuration is committed.

----End

Checking the Configurations

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 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.
- AS external routes are imported to the NSSA.

1.1.5.16 Configuring OSPF Local MT

By configuring local multicast topology (MT), you can create a routing table for multicast traffic when multicast and a Multiprotocol Label Switching (MPLS) traffic engineering (TE) tunnel are configured on a network.

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. The TE tunnel interface on the Router sends multicast Join packets over a unicast route to the multicast source address. The multicast Join packets are transparent to the Router through which the TE tunnel passes. As a result, the Router through which the TE tunnel passes cannot generate multicast forwarding entries.

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.

You can configure a routing policy for local MT to control the number of routing entries in an MIGP routing table and speed up 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 to 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 routing policy is configured for 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 the IP prefix:

  Run **local-mt filter-policy ip-prefix** *ip-prefix-name*

  A routing policy is configured for local MT.
- Based on the Route-Policy:

  Run **local-mt filter-policy route-policy** *route-policy-name*

  A routing policy is configured for local MT.
- Based on a route-filter:

  Run **local-mt filter-policy route-filter** *route-filter-name*

  A routing policy is configured for 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.

1.1.5.17 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** *soure-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.

1.1.5.18 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**.

1.1.5.18.1 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

1.1.5.18.2 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.

 Copyright © Huawei Technologies Co., Ltd.

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

1.1.5.18.3 (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** command can be run only in the VLANIF interface view.

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

1.1.5.18.4 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

1.1.5.19 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 1-36**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-36 OSPF IP FRR link protection

OSPF IP FRR is applicable to services that are sensitive to packet delay and packet loss.

The NE40E 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 1-37**, 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 1-37 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.

1.1.5.19.1 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:

1. Run **remote-lfa tunnel ldp** [**maximum-reachable-cost** *cost-value*]

   Remote LFA OSPF IP FRR is enabled.

2. (Optional) Run **remote-lfa available-tunnel-destination ip-prefix** *ip-prefix-name*

   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 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** *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 1-38**, 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 1-38 Solution of selecting a backup path for OSPF IP FRR

Figure 1-39 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 1-39 Maximum-cost (set by LDP-IGP synchronization) path first solution

Step 8 Run **commit**

The configuration is committed.

----End

1.1.5.19.2 (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

1.1.5.19.3 (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

1.1.5.19.4 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

1.1.5.20 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 used to ensure 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 switchover and system upgrade.

NOTE

GR includes the GR restarter and GR helper. NE40E 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 (Opaque LSAs). Therefore, before configuring OSPF 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.

1.1.5.21 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 a local device receives OSPF packets from a remote device, the local device discards the packets if the authentication passwords carried in these packets do not match the local one, which protects the local device from potential attacks.

In terms of the packet type, the authentication is classified as follows:

- Area authentication

  Area authentication is configured in the OSPF area view and applies to packets received by all interfaces in the OSPF area.

- Interface authentication

  Interface authentication is configured in the interface view and applies to all packets received by 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.

1.1.5.21.1 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

NOTE

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 process 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

1.1.5.21.2 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

1.1.5.21.3 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

1.1.5.22 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.

1.1.5.23 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.

1.1.5.24 Configuring Micro-Segment CAR for OSPF

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 packets of non-OSPF neighbors 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

1.1.5.25 Configuring the Threshold Alarm Function for the Number of OSPF Neighbors

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

1.1.5.26 Maintaining OSPF

Maintaining OSPF involves resetting OSPF and clearing OSPF statistics.

1.1.5.26.1 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

1.1.5.26.2 Resetting OSPF

Restarting OSPF can reset OSPF.

Context

NOTICE

The OSPF neighbor relationships will be torn down if you run the **reset ospf** command to reset OSPF connections. 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

1.1.5.26.3 Disabling OSPF Memory Overload Control

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

1.1.5.26.4 Disabling OSPF CPU Overload Control

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

1.1.5.27 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 1-10**.

Table 1-10 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.	**1.1.5.27.6 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.	**1.1.5.27.4 Configuring Fast Convergence for a Multi-Area Adjacency Interface**
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.	**1.1.5.27.7 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.	**1.1.5.27.5 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.	**1.1.5.12 Configuring OSPF Flush LSA Source Tracing**

 Copyright © Huawei Technologies Co., Ltd.

OSPF Feature	Description	Related Configuration Links
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**.

1.1.5.27.1 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

1.1.5.27.2 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

1.1.5.27.3 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

1.1.5.27.4 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

1.1.5.27.5 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.

NOTE

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

1.1.5.27.6 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

1.1.5.27.7 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

1.1.5.27.8 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

1.1.5.27.9 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

1.1.5.27.10 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

1.1.5.28 Configuration Examples for OSPF

This section provides several OSPF configuration examples.

1.1.5.28.1 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 1-40**, 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 1-40 Networking for configuring basic OSPF functions

NOTE

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

 Copyright © Huawei Technologies Co., Ltd.

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-0.0.0.0] network 192.168.0.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.0] quit
[*DeviceA-ospf-1] area 1
[*DeviceA-ospf-1-area-0.0.0.1] network 192.168.1.0 0.0.0.255
[*DeviceA-ospf-1-area-0.0.0.1] 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-0.0.0.0] network 192.168.0.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] quit
[*DeviceB-ospf-1] area 2
[*DeviceB-ospf-1-area-0.0.0.2] network 192.168.2.0 0.0.0.255
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*DeviceB-ospf-1-area-0.0.0.2] 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-0.0.0.1] network 192.168.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.1] network 172.16.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.1] commit
[~DeviceC-ospf-1-area-0.0.0.1] quit
```

Configure DeviceD.

```
[~DeviceD] router id 4.4.4.4
[*DeviceD] ospf 1
[*DeviceD-ospf-1] area 2
[*DeviceD-ospf-1-area-0.0.0.2] network 192.168.2.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0.2] network 172.17.1.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0.2] commit
[~DeviceD-ospf-1-area-0.0.0.2] quit
```

Configure DeviceE.

```
[~DeviceE] router id 5.5.5.5
[~DeviceE] ospf 1
[*DeviceE-ospf-1] area 1
[*DeviceE-ospf-1-area-0.0.0.1] network 172.16.1.0 0.0.0.255
[*DeviceE-ospf-1-area-0.0.0.1] commit
[~DeviceE-ospf-1-area-0.0.0.1] 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
```

NOTE

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 interface
         OSPF Process 1 with Router ID 1.1.1.1
                Neighbors

 Area 0.0.0.0 interface 192.168.0.1(GigabitEthernet0/1/0)'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(GigabitEthernet0/2/0)'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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
  authentication-mode hmac-sha256 1 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%#
 area 0.0.0.1
  network 192.168.1.0 0.0.0.255
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
router id 2.2.2.2
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

```
 authentication-mode hmac-sha256 1 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%#
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
router id 4.4.4.4
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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
```

1.1.5.28.2 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 1-41**, 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 1-41 Networking for configuring an OSPF virtual links

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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 (GigabitEthernet0/1/0)
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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
```

1.1.5.28.3 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 1-42**, 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 1-42 Networking for configuring an OSPF stub area

NOTE

Interfaces 1 and 2 in this example represent GE0/1/0 and GE0/2/0, 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

 Copyright © Huawei Technologies Co., Ltd.

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 **1.1.5.28.1 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          0.0.0.1    1     192.168.1.1    ABR
 Inter-area  4.4.4.4          0.0.0.1    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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
```

 Copyright © Huawei Technologies Co., Ltd.

```
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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
```

 Copyright © Huawei Technologies Co., Ltd.

1.1.5.28.4 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 1-43**, 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 1-43 Networking for configuring an OSPF NSSA

NOTE

Interfaces 1 through 3 in this example represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 **1.1.5.3 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 1-43**. For configuration details, see **Configuration Files** in this section.

Step 2 Configure basic OSPF functions.

1.1.5.3 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
```

NOTE

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
 10.0.0.0/8        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
10.0.0.0/8      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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.2.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.4.2 255.255.255.0
#
ospf 1
 area 0.0.0.0
  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
```

 Copyright © Huawei Technologies Co., Ltd.

```
 nssa
#
return
```

- DeviceC configuration file

```
#
 sysname DeviceC
#
router id 3.3.3.3
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
#
ospf 1
 area 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.3.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.5.28.5 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 1-44**, 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 1-44 Networking for configuring OSPF DR election

NOTE

Interface 1 in this example represents GE 0/1/0.

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 ( GE0/1/0 )'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 ( GE0/1/0 )'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 ( GE0/1/0 )'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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ospf dr-priority 100
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure DeviceB.

```
[~DeviceB] interface gigabitethernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] ospf dr-priority 0
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] quit
```

Configure DeviceC.

```
[~DeviceC] interface gigabitethernet 0/1/0
[~DeviceC-GigabitEthernet0/1/0] ospf dr-priority 2
[*DeviceC-GigabitEthernet0/1/0] commit
[~DeviceC-GigabitEthernet0/1/0] 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 ( GE0/1/0 )'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 ( GE0/1/0 )'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 ( GE0/1/0 )'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 ( GE0/1/0 )'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 0.0.0.0 interface 192.168.1.4 ( GE0/1/0 )'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 0.0.0.0 interface 192.168.1.4 ( GE0/1/0 )'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
 GigabitEthernet0/1/0   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
GigabitEthernet0/1/0   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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

1.1.5.28.6 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 1-45**:

- 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 1-45 Networking for configuring OSPF load balancing

NOTE

Interfaces 1 through 4 in this example represent GE0/1/0, GE0/2/0, GE0/3/0, and GE0/1/8, 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. Configure load balancing on DeviceA.
3. Set the priority for equal-cost routes on DeviceA.
4. 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 **1.1.5.28.1 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        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0       D        127.0.0.1        GigabitEthernet0/1/0
       10.1.1.2/32  Direct 0    0       D        10.1.1.2        GigabitEthernet0/1/0
       10.1.2.0/24  Direct 0    0       D        10.1.2.1        GigabitEthernet0/2/0
       10.1.2.1/32  Direct 0    0       D        127.0.0.1        GigabitEthernet0/2/0
       10.1.2.2/32  Direct 0    0       D        10.1.2.2        GigabitEthernet0/2/0
       10.1.3.0/24  Direct 0    0       D        10.1.2.1        GigabitEthernet0/3/0
       10.1.3.1/32  Direct 0    0       D        127.0.0.1        GigabitEthernet0/3/0
       10.1.3.2/32  Direct 0    0       D        10.1.2.2        GigabitEthernet0/3/0
    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        GigabitEthernet0/1/0
    192.168.1.0/24  OSPF   10   2       D        10.1.2.2        GigabitEthernet0/2/0
    192.168.2.0/24  OSPF   10   2       D        10.1.2.2        GigabitEthernet0/3/0
     172.17.1.0/24  OSPF   10   3       D        10.1.1.2        GigabitEthernet0/1/0
                    OSPF   10   3       D        10.1.2.2        GigabitEthernet0/2/0
                    OSPF   10   3       D        10.1.3.2        GigabitEthernet0/3/0
```

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        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0       D        127.0.0.1        GigabitEthernet0/1/0
       10.1.1.2/32  Direct 0    0       D        10.1.1.2        GigabitEthernet0/1/0
       10.1.2.0/24  Direct 0    0       D        10.1.2.1        GigabitEthernet0/2/0
       10.1.2.1/32  Direct 0    0       D        127.0.0.1        GigabitEthernet0/2/0
       10.1.2.2/32  Direct 0    0       D        10.1.2.2        GigabitEthernet0/2/0
       10.1.3.0/24  Direct 0    0       D        10.1.2.1        GigabitEthernet0/3/0
       10.1.3.1/32  Direct 0    0       D        127.0.0.1        GigabitEthernet0/3/0
       10.1.3.2/32  Direct 0    0       D        10.1.2.2        GigabitEthernet0/3/0
    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        GigabitEthernet0/1/0
    192.168.1.0/24  OSPF   10   2       D        10.1.2.2        GigabitEthernet0/2/0
    192.168.2.0/24  OSPF   10   2       D        10.1.2.2        GigabitEthernet0/3/0
     172.17.1.0/24  OSPF   10   3       D        10.1.1.2        GigabitEthernet0/1/0
                    OSPF   10   3       D        10.1.2.2        GigabitEthernet0/2/0
```

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        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0        D        127.0.0.1        GigabitEthernet0/1/0
       10.1.1.2/32  Direct 0    0        D        10.1.1.2        GigabitEthernet0/1/0
       10.1.2.0/24  Direct 0    0        D        10.1.2.1        GigabitEthernet0/2/0
       10.1.2.1/32  Direct 0    0        D        127.0.0.1        GigabitEthernet0/2/0
       10.1.2.2/32  Direct 0    0        D        10.1.2.2        GigabitEthernet0/2/0
       10.1.3.0/24  Direct 0    0        D        10.1.2.1        GigabitEthernet0/3/0
       10.1.3.1/32  Direct 0    0        D        127.0.0.1        GigabitEthernet0/3/0
       10.1.3.2/32  Direct 0    0        D        10.1.2.2        GigabitEthernet0/3/0
    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        GigabitEthernet0/1/0
    192.168.1.0/24  OSPF   10   2        D        10.1.2.2        GigabitEthernet0/2/0
    192.168.2.0/24  OSPF   10   2        D        10.1.2.2        GigabitEthernet0/3/0
     172.17.1.0/24  OSPF   10   3        D        10.1.2.2        GigabitEthernet0/2/0
                    OSPF   10   3        D        10.1.3.2        GigabitEthernet0/3/0
```

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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
#
ospf 1 router-id 2.2.2.2
 area 0.0.0.0
  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 GigabitEthernet0/1/0
 ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.1.1 255.255.255.0
#
ospf 1 router-id 3.3.3.3
 area 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.2.1 255.255.255.0
#
ospf 1 router-id 4.4.4.4
 area 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
#
interface GigabitEthernet0/1/8
```

```
 undo shutdown
 ip address 172.17.1.1 255.255.255.0
#
ospf 1 router-id 5.5.5.5
 area 0.0.0.0
  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
```

1.1.5.28.7 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 1-46**, OSPF runs on the four devices, which belong to the same OSPF area.

Figure 1-46 Networking for configuring OSPF fast convergence

NOTE

Interface 1 in this example represents GE0/1/0.

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 **1.1.5.28.1 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ospf timer dead 30
[*DeviceA-GigabitEthernet0/1/0] 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-GigabitEthernet0/1/0] ospf smart-discover
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 (GE0/3/1)
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

 Copyright © Huawei Technologies Co., Ltd.

```
 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 GigabitEthernet0/1/0
 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
```

1.1.5.28.8 Example for Configuring OSPF Local MT

This section provides an example for configuring OSPF local multicast topology (MT).

Networking Requirements

When multicast and an Interior Gateway Protocol (IGP) Shortcut-enabled Multiprotocol Label Switching (MPLS) traffic engineering (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. 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 1-47**, 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 0/2/0. Tunnel 10 on DeviceB sends multicast Join packets over a unicast route to the multicast source address. The multicast Join packets are transparent to DeviceC through which the TE tunnel passes. 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 1-47 Configuring local OSPF MT

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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 1-11** lists the IP addresses of interfaces on the Router.

Table 1-11 IP address of each interface

Device	IP Address of Loopback 0
DeviceA	1.1.1.1/32
DeviceB	2.2.2.2/32
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 1-47 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 0/2/0
[~DeviceA-GigabitEthernet0/2/0] pim sm
[*DeviceA-GigabitEthernet0/2/0] quit
[*DeviceA] interface Gigabitethernet 0/1/0
[*DeviceA-GigabitEthernet0/1/0] pim sm
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

NOTE

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 0/1/0 of DeviceA.

```
[~DeviceA] interface Gigabitethernet 0/1/0
[~DeviceA-GigabitEthernet0/1/0] igmp enable
[*DeviceA-GigabitEthernet0/1/0] igmp version 3
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 Gigabitethernet 0/1/0
```

```
[*DeviceD-pim] c-rp Gigabitethernet 0/1/0
[*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: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0

 00002. (192.168.3.9, 224.31.31.31)
      Uptime: 15:03:04
      Upstream Interface: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0

 00003. (192.168.3.10, 224.31.31.31)
      Uptime: 15:03:04
      Upstream Interface: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0
```

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 0/2/0
[*DeviceB-GigabitEthernet0/2/0] mpls
[*DeviceB-GigabitEthernet0/2/0] mpls te
[*DeviceB-GigabitEthernet0/2/0] mpls rsvp-te
[*DeviceB-GigabitEthernet0/2/0] commit
[*DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] mpls
[*DeviceC-GigabitEthernet0/1/0] mpls te
[*DeviceC-GigabitEthernet0/1/0] mpls rsvp-te
[*DeviceC-GigabitEthernet0/1/0] commit
```

```
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface Gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] mpls
[*DeviceC-GigabitEthernet0/2/0] mpls te
[*DeviceC-GigabitEthernet0/2/0] mpls rsvp-te
[*DeviceC-GigabitEthernet0/2/0] commit
[*DeviceC-GigabitEthernet0/2/0] 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 0/2/0
[*DeviceD-GigabitEthernet0/2/0] mpls
[*DeviceD-GigabitEthernet0/2/0] mpls te
[*DeviceD-GigabitEthernet0/2/0] mpls rsvp-te
[*DeviceD-GigabitEthernet0/2/0] commit
[~DeviceD-GigabitEthernet0/2/0] 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        GigabitEthernet0/2/0
      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        GigabitEthernet0/1/0
        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        GigabitEthernet0/2/0
        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        GigabitEthernet0/2/0
                    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     GigabitEthernet0/2/0
      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: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0

 00002. (192.168.3.9, 224.31.31.31)
      Uptime: 00:00:19
      Upstream Interface: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0

 00003. (192.168.3.10, 224.31.31.31)
      Uptime: 00:00:19
      Upstream Interface: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0
```

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 GigabitEthernet0/2/0
 undo shutdown
 ip address 10.0.0.1 255.255.255.0
 pim sm
#
interface GigabitEthernet0/1/0
 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 0.0.0.0
  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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.0.0.2 255.255.255.0
 pim sm
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.0.1.1 255.255.255.0
 pim sm
 mpls
 mpls te
 mpls rsvp-te
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.0.3.1 255.255.255.0
 pim sm
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 C-RP GigabitEthernet0/1/0
#
return
```

- Device E configuration file

```
#
 sysname DeviceE
#
 router id 5.5.5.5
#
 multicast routing-enable
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.0.3.2 255.255.255.0
 pim sm
#
interface GigabitEthernet0/2/0
 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
```

1.1.5.28.9 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 1-48**:

- 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 1-48 Networking for configuring OSPF IP FRR

NOTE

Interfaces 1 through 4 in this example represent GE 0/1/0, GE 0/2/0, GE 0/3/0, and GE 0/1/1, respectively.

Precautions

When configuring OSPF IP FRR, note the following rules:

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

associated with the link status of an interface so that link 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	GE0/1/0	10.1.1.1/24
		GE0/2/0	10.1.2.1/24
		GE0/3/0	10.1.3.1/24
DeviceA	2.2.2.2	GE0/1/0	10.1.1.2/24
		GE0/2/0	10.2.1.2/24
DeviceN	3.3.3.3	GE0/1/0	10.1.3.2/24
		GE0/2/0	10.2.3.2/24
DeviceE	4.4.4.4	GE0/1/0	10.2.1.1/24
		GE0/2/0	10.1.2.2/24
		GE0/3/0	10.2.3.1/24
		GE0/1/1	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 gigabitethernet0/1/0
[*DeviceS-GigabitEthernet0/1/0] ospf cost 10
[*DeviceS-GigabitEthernet0/1/0] quit
[*DeviceS] interface gigabitethernet0/2/0
[*DeviceS-GigabitEthernet0/2/0] ospf cost 15
[*DeviceS-GigabitEthernet0/2/0] quit
[*DeviceS] interface gigabitethernet0/3/0
[*DeviceS-GigabitEthernet0/3/0] ospf cost 10
[*DeviceS-GigabitEthernet0/3/0] quit
[*DeviceS] commit
```

Configure DeviceA.

```
[~DeviceA] interface gigabitethernet0/2/0
[*DeviceA-GigabitEthernet0/2/0] ospf cost 15
[*DeviceA-GigabitEthernet0/2/0] quit
[*DeviceA] commit
```

Configure DeviceN.

```
[~DeviceN] interface gigabitethernet0/2/0
[*DeviceN-GigabitEthernet0/2/0] ospf cost 10
[*DeviceN-GigabitEthernet0/2/0] quit
[*DeviceN] commit
```

Step 5 Block FRR on a specified interface of DeviceS.

```
[~DeviceS] interface gigabitethernet0/1/0
[*DeviceS-GigabitEthernet0/1/0] ospf frr block
[*DeviceS-GigabitEthernet0/1/0] 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] 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  :       GE0/2/0
 Backup Nexthop : 10.1.3.2        Backup Interface : GE0/3/0
 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 GigabitEthernet0/1/0
 ip address 10.1.1.1 255.255.255.0
 ospf frr block
 ospf cost 10
#
interface GigabitEthernet0/2/0
 ip address 10.1.2.1 255.255.255.0
 ospf cost 15
#
interface GigabitEthernet0/3/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 ip address 10.1.1.2 255.255.255.0
 ospf cost 10
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 ip address 10.1.3.2 255.255.255.0
 ospf cost 10
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 ip address 10.2.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 ip address 10.2.3.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 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
```

1.1.5.28.10 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 1-49**, 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 1-49 Networking for configuring BFD for OSPF

NOTE

Interfaces 1 through 3 in this example represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 interface
         OSPF Process 1 with Router ID 1.1.1.1
              Neighbors

 Area 0.0.0.0 interface 3.3.3.1 ( GE0/2/0 )'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 ( GE0/1/0 )'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 0.0.0.0 interface 1.1.1.1(GE0/1/0)'s BFD Sessions

 NeighborId:2.2.2.2          AreaId:0.0.0.0          Interface:GE0/1/0
 BFDState:up                 rx    :1000             tx       :1000
 Multiplier:3                BFD Local Dis:0         LocalIpAdd:1.1.1.1
 RemoteIpAdd:1.1.1.2         Diagnostic Info:0

  Area 0.0.0.0 interface 3.3.3.1(GE0/2/0)'s BFD Sessions

 NeighborId:3.3.3.3          AreaId:0.0.0.0          Interface:GE0/2/0
 BFDState:up                 rx    :1000             tx       :1000
 Multiplier:3                BFD Local Dis:0         LocalIpAdd:3.3.3.1
 RemoteIpAdd:3.3.3.2         Diagnostic Info:0
```

Step 4 Configure BFD on an interface.

Configure BFD on GigabitEthernet 0/2/0 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 0/2/0
[*DeviceA-GigabitEthernet0/2/0] ospf bfd enable
[*DeviceA-GigabitEthernet0/2/0] ospf bfd min-tx-interval 500 min-rx-interval 500 detect-multiplier 4
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] quit
```

Configure BFD on GigabitEthernet 0/2/0 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 0/2/0
[*DeviceB-GigabitEthernet0/2/0] ospf bfd enable
[*DeviceB-GigabitEthernet0/2/0] ospf bfd min-tx-interval 500 min-rx-interval 500 detect-multiplier 4
[*DeviceB-GigabitEthernet0/2/0] commit
[~DeviceB-GigabitEthernet0/2/0] 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 0.0.0.0 interface 3.3.3.2(GE0/2/0)'s BFD Sessions

 NeighborId:1.1.1.1          AreaId:0.0.0.0          Interface: GigabitEthernet0/2/0
 BFDState:up                 rx    :500              tx       :500
 Multiplier:4                BFD Local Dis:0         LocalIpAdd:3.3.3.2
 RemoteIpAdd:3.3.3.1         Diagnostic Info:0

  Area 0.0.0.0 interface 2.2.2.2(GE0/1/0)'s BFD Sessions

 NeighborId:3.3.3.3          AreaId:0.0.0.0          Interface: GE0/1/0
 BFDState:up                 rx    :1000             tx       :1000
```

```
Multiplier:3              BFD Local Dis:0        LocalIpAdd:2.2.2.2
RemoteIpAdd:2.2.2.1        Diagnostic Info:0
```

Step 5 Verify the configuration.

Run the **shutdown** command on GigabitEthernet 0/2/0 of DeviceB to simulate a primary link failure.

```
[~DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] shutdown
[*DeviceB-GigabitEthernet0/2/0] 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 1.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ip address 2.2.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
#
ospf 1
 bfd all-interfaces enable
 area 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 2.2.2.1 255.255.255.0
 ospf bfd enable
#
ospf 1
 bfd all-interfaces enable
 area 0.0.0.0
  network 1.1.1.0 0.0.0.255
  network 2.2.2.0 0.0.0.255
#
return
```

1.1.5.28.11 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 1-50**, 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 1-50 Networking for configuring OSPF-BGP synchronization

NOTE

Interfaces 1 through 3 in this example represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 **1.1.5.28.1 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] ospf cost 2
[*DeviceC-GigabitEthernet0/1/0] quit
[~DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ospf cost 2
[*DeviceC-GigabitEthernet0/2/0] commit
[~DeviceC-GigabitEthernet0/2/0] quit
```

NOTE

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. BGP learns the route to 10.1.3.0, and the outbound interface is GE 0/1/0.

```
[~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        GigabitEthernet0/1/0
     4.4.4.0/24         BGP      255  0       RD    4.4.4.4         GigabitEthernet0/1/0
     4.4.4.4/32         OSPF     10   3       D     10.1.1.2        GigabitEthernet0/1/0
     5.5.5.0/24         BGP      255  0       RD    10.2.1.2        GigabitEthernet0/1/0
    10.1.1.0/30         Direct   0    0       D     10.1.1.1        GigabitEthernet0/1/0
    10.1.1.3/32         Direct   0    0       D     10.1.1.1        GigabitEthernet0/1/0
    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        GigabitEthernet0/1/0
    10.1.1.2/32         Direct   0    0       D     10.1.1.2        GigabitEthernet0/1/0
    10.1.1.255/32       Direct   0    0       D     10.1.1.2        GigabitEthernet0/1/0
    10.1.2.0/30         Direct   0    0       D     10.1.2.1        GigabitEthernet0/2/0
    10.1.2.3/32         Direct   0    0       D     10.1.2.1        GigabitEthernet0/2/0
    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        GigabitEthernet0/2/0
    10.1.2.255/32       Direct   0    0       D     10.1.2.2        GigabitEthernet0/2/0
   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
  10.3.1.0/30       OSPF    10   2       D     10.1.1.2        GigabitEthernet0/1/0
    10.1.3.1/32         BGP      255  0       RD    4.4.4.4         GigabitEthernet0/1/0
    10.1.4.0/30         OSPF     10   3       D     10.1.1.2        GigabitEthernet0/1/0
                        OSPF     10   3       D     10.1.2.2        GigabitEthernet0/2/0
    10.1.4.1/32         BGP      255  0       RD    4.4.4.4         GigabitEthernet0/1/0
    10.2.1.0/30         BGP      255  0       RD    4.4.4.4         GigabitEthernet0/1/0
    10.2.1.2/32         BGP      255  0       RD    4.4.4.4         GigabitEthernet0/1/0
    10.3.1.0/30         BGP      255  0       RD    4.4.4.4         GigabitEthernet0/1/0
    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        GigabitEthernet0/1/0
      4.4.4.0/24         BGP      255  0           RD  10.1.3.2        GigabitEthernet0/2/0
      4.4.4.4/32         OSPF     10   2           D   10.1.3.2        GigabitEthernet0/2/0
      5.5.5.0/24         BGP      255  0           RD  10.2.1.2        GigabitEthernet0/2/0
     10.1.1.0/30         Direct   0    0           D   10.1.1.2        GigabitEthernet0/1/0
     10.1.1.3/32         Direct   0    0           D   10.1.1.2        GigabitEthernet0/1/0
     10.1.1.1/32         Direct   0    0           D   10.1.1.1        GigabitEthernet0/1/0
     10.1.1.255/32       Direct   0    0           D   10.1.1.1        GigabitEthernet0/1/0
     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        GigabitEthernet0/1/0
     10.1.3.0/30         Direct   0    0           D   10.1.3.1        GigabitEthernet0/2/0
     10.1.3.3/32         Direct   0    0           D   10.1.3.1        GigabitEthernet0/2/0
     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        GigabitEthernet0/2/0
     10.1.3.255/32       Direct   0    0           D   10.1.3.2        GigabitEthernet0/2/0
    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        GigabitEthernet0/2/0
     10.1.4.1/32         BGP      255  0           RD  10.1.3.2        GigabitEthernet0/2/0
     10.2.1.0/30         BGP      255  0           RD  10.1.3.2        GigabitEthernet0/2/0
     10.2.1.2/32         BGP      255  0           RD  10.1.3.2        GigabitEthernet0/2/0
   10.3.1.0/30       BGP     255  0           RD  10.1.3.2        GigabitEthernet0/2/0
     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 0/2/0. 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 GigabitEthernet0/2/0.

```
[~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        GigabitEthernet0/2/0
       4.4.4.0/24  BGP    255  0           RD  4.4.4.4         GigabitEthernet0/2/0
       4.4.4.4/32  OSPF   10   4           D   10.1.2.2        GigabitEthernet0/2/0
       5.5.5.0/24  BGP    255  0           RD  10.2.1.2        GigabitEthernet0/2/0
      10.1.1.0/30  Direct 0    0           D   10.1.1.1        GigabitEthernet0/1/0
      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        GigabitEthernet0/1/0
      10.1.2.0/30  Direct 0    0           D   10.1.2.1        GigabitEthernet0/2/0
      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        GigabitEthernet0/2/0
     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        GigabitEthernet0/1/0
      10.1.3.1/32  BGP    255  0           RD  4.4.4.4         GigabitEthernet0/2/0
      10.1.4.0/30  OSPF   10   3           D   10.1.2.2        GigabitEthernet0/2/0
      10.1.4.1/32  BGP    255  0           RD  4.4.4.4         GigabitEthernet0/2/0
      10.2.1.0/30  BGP    255  0           RD  4.4.4.4         GigabitEthernet0/2/0
      10.2.1.2/32  BGP    255  0           RD  4.4.4.4         GigabitEthernet0/2/0
     10.3.1.0/30   BGP    255  0           RD  4.4.4.4         GigabitEthernet0/2/0
```

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  OSPF   10   65536       D  10.1.1.1        GigabitEthernet0/1/0
       2.2.2.2/32  Direct 0    0           D  127.0.0.1       InLoopBack0
```

```
     4.4.4.4/32  OSPF   10   65536       D  10.1.3.2        GigabitEthernet0/2/0
     10.1.1.0/30  Direct 0    0           D  10.1.1.2        GigabitEthernet0/1/0
     10.1.1.1/32  Direct 0    0           D  10.1.1.1        GigabitEthernet0/1/0
     10.1.1.2/32  Direct 0    0           D  127.0.0.1        InLoopBack0
     10.1.2.0/30  OSPF   10   65536       D  10.1.1.1        GigabitEthernet0/1/0
     10.1.3.0/30  Direct 0    0           D  10.1.3.1        GigabitEthernet0/2/0
     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        GigabitEthernet0/2/0
     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   65536       D  10.1.3.2        GigabitEthernet0/2/0
     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        GigabitEthernet0/1/0
       4.4.4.0/24  BGP    255  0           RD  10.1.3.2        GigabitEthernet0/2/0
       4.4.4.4/32  OSPF   10   2           D   10.1.3.2        GigabitEthernet0/2/0
       5.5.5.0/24  BGP    255  0           RD  10.2.1.2        GigabitEthernet0/2/0
       10.1.1.0/30  Direct 0    0           D   10.1.1.2        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0           D   10.1.1.1        GigabitEthernet0/1/0
       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        GigabitEthernet0/1/0
       10.1.3.0/30  Direct 0    0           D   10.1.3.1        GigabitEthernet0/2/0
       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        GigabitEthernet0/2/0
       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        GigabitEthernet0/2/0
       10.1.4.1/32  BGP    255  0           RD  10.1.3.2        GigabitEthernet0/2/0
       10.2.1.0/30  BGP    255  0           RD  10.1.3.2        GigabitEthernet0/2/0
       10.2.1.2/32  BGP    255  0           RD  10.1.3.2        GigabitEthernet0/2/0
       10.3.1.0/30  BGP    255  0           RD  10.1.3.2        GigabitEthernet0/2/0
```

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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.252
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.252
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.252
 ospf cost 2
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.2 255.255.255.252
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.3.2 255.255.255.252
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
```

```
 undo shutdown
 ip address 10.2.1.2 255.255.255.252
#
interface GigabitEthernet0/2/0
 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
```

1.1.5.28.12 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 1-51**, 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 1-51 Routing loop detection for routes imported from BGP to OSPF

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ip address 10.12.1.1 24
[*DeviceA-GigabitEthernet0/1/0] 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 0.0.0.0 interface 10.34.1.3 (GigabitEthernet0/1/0)'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 0.0.0.0 interface 10.34.1.4 (GigabitEthernet0/2/0)'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: GigabitEthernet0/1/0
 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:GigabitEthernet0/2/0
 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           : GigabitEthernet0/2/0
 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 (1.11.1.1)
 Route Duration: 1d00h10m02s
 Relay IP Nexthop: 10.12.1.1
 Relay IP Out-Interface: GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.12.1.1 255.255.255.0
#
bgp 100
 router-id 1.11.1.1
 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
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.12.1.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 10.24.1.2 255.255.255.0
#
interface GigabitEthernet0/1/0
 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
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.34.1.3 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
#
return
```

- DeviceD configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.34.1.4 255.255.255.0
#
interface GigabitEthernet0/1/0
 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
#
return
```

1.1.5.28.13 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 1-52**, 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 1-52 Networking for configuring OSPF multi-area adjacency

NOTE

Interfaces 1 and 2 in this example represent GE0/1/0 and GE0/2/0, 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 0/1/0
```

```
[*DeviceA-GigabitEthernet0/1/0] ospf enable 1 area 1
[*DeviceA-GigabitEthernet0/1/0] 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 0/2/0
[*DeviceA-GigabitEthernet0/2/0] ospf enable 1 area 2
[*DeviceA-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ospf enable 1 area 1
[*DeviceB-GigabitEthernet0/1/0] 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 0/2/0
[*DeviceB-GigabitEthernet0/2/0] ospf enable 1 area 2
[*DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] ospf enable 1 area 2
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ospf enable 1 area 2
[*DeviceC-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] ospf enable 1 area 2
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] ospf enable 1 area 2
[*DeviceD-GigabitEthernet0/2/0] 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          GigabitEthernet0/1/0        10.1.1.2             Full
 0.0.0.2          GigabitEthernet0/2/0        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 0/2/0.

```
[~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        GigabitEthernet0/2/0
       10.1.1.0/24  Direct  0    0           D   10.1.1.1        GigabitEthernet0/1/0
       10.1.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.1.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
       10.1.2.0/24  Direct  0    0           D   10.1.2.1        GigabitEthernet0/2/0
       10.1.2.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
     10.1.2.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
       10.1.3.0/24  OSPF    10   3           D   10.1.2.2        GigabitEthernet0/2/0
       10.1.4.0/24  OSPF    10   2           D   10.1.2.2        GigabitEthernet0/2/0
      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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ospf enable multi-area 2
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] commit
```

Configure DeviceB.

```
[~DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ospf enable multi-area 2
[*DeviceB-GigabitEthernet0/1/0] 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        GigabitEthernet0/1/0       10.1.1.2            Full
 0.0.0.2        GigabitEthernet0/1/0       10.1.1.2(M)          Full
 0.0.0.2        GigabitEthernet0/2/0       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 0/1/0.

```
[~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        GigabitEthernet0/1/0
       10.1.1.0/24  Direct  0    0           D   10.1.1.1        GigabitEthernet0/1/0
       10.1.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.1.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
       10.1.2.0/24  Direct  0    0           D   10.1.2.1        GigabitEthernet0/2/0
       10.1.2.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
     10.1.2.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
       10.1.3.0/24  OSPF    10   2           D   10.1.1.2        GigabitEthernet0/1/0
       10.1.4.0/24  OSPF    10   2           D   10.1.2.2        GigabitEthernet0/2/0
      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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.0
 ospf enable 1 area 0.0.0.2
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.2 255.255.255.0
 ospf enable 1 area 0.0.0.2
#
interface GigabitEthernet0/2/0
 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
```

1.1.5.28.14 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 1-53**, 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 1-53 Networking for configuring an OSPF sham link

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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 GigabitEthernet0/2/0
[~CE1-GigabitEthernet0/2/0] ip address 10.1.1.1 24
[*CE1-GigabitEthernet0/2/0] quit
[*CE1] interface GigabitEthernet0/1/0
[*CE1-GigabitEthernet0/1/0] ip address 172.16.1.1 24
[*CE1-GigabitEthernet0/1/0] 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 GigabitEthernet0/1/0
[~RT0-GigabitEthernet0/1/0] ip address 10.1.1.2 24
[*RT0-GigabitEthernet0/1/0] quit
[*RT0] interface GigabitEthernet0/2/0
[*RT0-GigabitEthernet0/2/0] ip address 10.2.1.1 24
[*RT0-GigabitEthernet0/2/0] 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 GigabitEthernet0/2/0
[~CE2-GigabitEthernet0/2/0] ip address 10.2.1.2 24
```

```
[*CE2-GigabitEthernet0/2/0] quit
[*CE2] interface GigabitEthernet0/1/0
[*CE2-GigabitEthernet0/1/0] ip address 172.16.2.1 24
[*CE2-GigabitEthernet0/1/0] 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 GigabitEthernet0/2/0
[*PE1-GigabitEthernet0/2/0] ip address 10.10.1.1 24
[*PE1-GigabitEthernet0/2/0] mpls
[*PE1-GigabitEthernet0/2/0] mpls ldp
[*PE1-GigabitEthernet0/2/0] 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 vpnv4
[*PE1-bgp-af-vpnv4] peer 3.3.3.9 enable
[*PE1-bgp-af-vpnv4] 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 GigabitEthernet0/1/0
[*P-GigabitEthernet0/1/0] ip address 10.10.1.2 24
[*P-GigabitEthernet0/1/0] mpls
[*P-GigabitEthernet0/1/0] mpls ldp
[*P-GigabitEthernet0/1/0] quit
[*P] interface GigabitEthernet0/2/0
[*P-GigabitEthernet0/2/0] ip address 10.20.1.1 24
[*P-GigabitEthernet0/2/0] mpls
[*P-GigabitEthernet0/2/0] mpls ldp
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*P-GigabitEthernet0/2/0] quit
[*P] ospf
[*P-ospf-1] area 0
[*P-ospf-1-area-0.0.0.0] network 2.2.2.9 0.0.0.0
[*P-ospf-1-area-0.0.0.0] network 10.10.1.0 0.0.0.255
[*P-ospf-1-area-0.0.0.0] network 10.20.1.0 0.0.0.255
[*P-ospf-1-area-0.0.0.0] 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 GigabitEthernet0/2/0
[*PE2-GigabitEthernet0/2/0] ip address 10.20.1.2 24
[*PE2-GigabitEthernet0/2/0] mpls
[*PE2-GigabitEthernet0/2/0] mpls ldp
[*PE2-GigabitEthernet0/2/0] quit
[*PE2] ospf
[*PE2-ospf-1] area 0
[*PE2-ospf-1-area-0.0.0.0] network 3.3.3.9 0.0.0.0
[*PE2-ospf-1-area-0.0.0.0] network 10.20.1.0 0.0.0.255
[*PE2-ospf-1-area-0.0.0.0] 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 0/1/0
[*PE1-GigabitEthernet0/1/0] ip binding vpn-instance vpn1
[*PE1-GigabitEthernet0/1/0] ip address 172.16.1.2 24
[*PE1-GigabitEthernet0/1/0] 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-0.0.0.0] network 172.16.1.0 0.0.0.255
[*PE1-ospf-100-area-0.0.0.0] 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 GigabitEthernet0/1/0
[*PE2-GigabitEthernet0/1/0] ip binding vpn-instance vpn1
[*PE2-GigabitEthernet0/1/0] ip address 172.16.2.2 24
[*PE2-GigabitEthernet0/1/0] 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
     10.1.1.0/24  OSPF    10   2           D   172.16.1.1     GigabitEthernet0/1/0
     10.2.1.0/24  OSPF    10   3           D   172.16.1.1     GigabitEthernet0/1/0
    172.16.1.0/24 Direct 0    0           D   172.16.1.2     GigabitEthernet0/1/0
    172.16.1.2/32 Direct 0    0           D   127.0.0.1      GigabitEthernet0/1/0
    172.16.2.0/24 OSPF    10   4           D   172.16.1.1     GigabitEthernet0/1/0
```

Step 4 Configure a sham link.

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 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 GigabitEthernet0/2/0
[~CE1-GigabitEthernet0/2/0] ospf cost 10
[*CE1-GigabitEthernet0/2/0] quit
[*CE1] commit
```

Configure CE2.

```
[~CE2] interface GigabitEthernet0/2/0
[~CE2-GigabitEthernet0/2/0] ospf cost 10
[*CE2-GigabitEthernet0/2/0] 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          GigabitEthernet0/2/0
       8.8.8.8/32  IBGP   255  2           RD  3.3.3.9          GigabitEthernet0/2/0
      10.1.1.0/24  OSPF   10   11          D   172.16.1.1       GigabitEthernet0/1/0
      10.2.1.0/24  OSPF   10   12          D   172.16.1.1       GigabitEthernet0/1/0
     172.16.1.0/24 Direct 0    0           D   172.16.1.2       GigabitEthernet0/1/0
     172.16.1.2/32 Direct 0    0           D   127.0.0.1        GigabitEthernet0/1/0
   172.16.1.255/32 Direct 0    0           D   127.0.0.1        GigabitEthernet0/1/0
     172.16.2.0/24 IBGP   255  0           RD  3.3.3.9          GigabitEthernet0/2/0
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
        5.5.5.5/32  O_ASE   150  1           D   172.16.1.2      GigabitEthernet0/1/0
        6.6.6.6/32  O_ASE   150  1           D   172.16.1.2      GigabitEthernet0/1/0
        8.8.8.8/32  OSPF    10   3           D   172.16.1.2      GigabitEthernet0/1/0
       10.1.1.0/24  Direct  0    0           D   10.1.1.1        GigabitEthernet0/2/0
       10.1.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
     10.1.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
       10.2.1.0/24  OSPF    10   11          D   10.1.1.2        GigabitEthernet0/2/0
      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      GigabitEthernet0/1/0
     172.16.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
   172.16.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     172.16.2.0/24  OSPF    10   3           D   172.16.1.2      GigabitEthernet0/1/0
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 0/1/0. 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: 0.0.0.0
  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
 8.8.8.8/32          3     Stub       172.16.1.2      10.2.1.2        0.0.0.0
 10.1.1.0/24         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
 6.6.6.6/32          1         Type2      3489661028  172.16.1.2      11.11.11.11
 5.5.5.5/32          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 GigabitEthernet0/1/0
 undo shutdown
 ip binding vpn-instance vpn1
 ip address 172.16.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 vpnv4
  policy vpn-target
  peer 3.3.3.9 enable
  #
```

```
 ipv4-family vpn-instance vpn1
  import-route direct
  import-route ospf 100
#
ospf 1
 area 0.0.0.0
  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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.10.1.2 255.255.255.0
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ip binding vpn-instance vpn1
 ip address 172.16.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 vpnv4
  policy vpn-target
  peer 1.1.1.9 enable
#
 ipv4-family vpn-instance vpn1
  import-route direct
  import-route ospf 100
#
ospf 1
 area 0.0.0.0
  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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
 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 172.16.1.0 0.0.0.255
  network 10.1.1.0 0.0.0.255
#
return
```

- CE2 configuration file

```
#
 sysname CE2
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.2.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.2.1.2 255.255.255.0
 ospf cost 10
#
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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.6 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.

1.1.6.1 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.

1.1.6.2 Configuration Precautions for OSPFv3

Feature Requirements

Table 1-12 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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
If the OSPFv3 dead timer is too short, OSPFv3 neighbor relationships may easily go down, interrupting services.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
If the OSPFv3 neighbor timeout period is too short, the neighbor relationship may easily go Down due to timeout, affecting services. You are advised to use the default hello timer and dead timer. Alternatively, you can change the value of the dead timer to a value greater than or equal to 40s.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
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. 2. Routes cannot be imported to NSSAs. 3. Summary routes cannot advertise redistribute information. 4. After a loop is removed, the loop cannot be automatically removed. You need to manually clear the alarm using a command. 5. If the redistribute IDs of different devices conflict, a loop alarm may be falsely reported. 6. Loop detection is not supported in the scenario where more than two re-advertisement nodes are configured. 7. Loops may fail to be detected in load balancing scenarios where inner-loop routes and non-loop routes are advertised by multiple nodes. 8. Self-healing is not supported when the original route source is withdrawn and a loop is triggered. 9. Route import between public and private networks is not supported. 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. 11. When the maximum cost is advertised on a loop, the apply cost command does not take effect. 12. After this function is enabled, an LSA needs to be added to each prefix to advertise extended TLVs, which greatly increases the memory usage.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.6.3 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.

1.1.6.3.1 Enabling OSPFv3

Creating an OSPFv3 process is a prerequisite for configuring OSPFv3 features. By creating an OSPFv3 process, you can specify the router ID for an OSPFv3 process.

Context

OSPFv3 supports multi-process. Multiple OSPFv3 processes running on one Device are differentiated by process IDs. You can set an OSPFv3 process ID when enabling OSPFv3. The process ID is only locally valid and does not affect the packet exchange with other Devices.

In the format of an IPv4 address, a router ID is a 32-bit unsigned integer that uniquely identifies one Device within an AS. The router ID of OSPFv3 must be set. Without a router ID, OSPFv3 cannot run properly.

When configuring the router ID, ensure that the router ID is unique in an AS. When multiple processes are enabled on the same Device, specify a unique router ID for each process.

Configure router IDs during network planning so that OSPFv3 can run properly.

Perform the following steps on the Device that runs OSPFv3:

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ospfv3** [*process-id*] [**vpn-instance** *vpnname*]

OSPFv3 is enabled, and the OSPFv3 view is displayed.

The description of an OSPFv3 process helps identify special processes by the **description** command.

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. 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:

- Reconfigure a 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

1.1.6.3.2 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

1.1.6.3.3 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

1.1.6.3.4 (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** *vpnname*]

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

1.1.6.3.5 (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

1.1.6.3.6 (Optional) Disabling an Interface from Receiving and Sending OSPFv3 Packets

Context

To prevent OSPFv3 routes from being received by other routers on the network and disable the local router from receiving updated routes, you can disable the local router's interface from receiving and sending 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*

An interface is disabled from receiving and sending OSPFv3 packets.

NOTE

The same interface in different processes can be disabled from sending and receiving OSPFv3 packets, but the **silent-interface** command is valid only for the OSPFv3 interface in the local process.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.6.3.7 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

1.1.6.4 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, HDLC or, LAPB, OSPFv3 defaults the network type to P2P.

You can change network types and configure OSPFv3 attributes 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 1-54 Flowchart for configuring OSPFv3 attributes on different types of networks

1.1.6.4.1 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**, the network type of serial interfaces is **p2p**, the network type of Frame-relay interfaces is **nbma**.

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

1.1.6.4.2 (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.

1.1.6.4.3 (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

1.1.6.4.4 (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

1.1.6.4.5 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

1.1.6.5 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**.

1.1.6.5.1 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

1.1.6.5.2 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

1.1.6.5.3 Setting the convergence priority for OSPFv3 routes

You can adjust and optimize route selection by setting the convergence priority for OSPFv3 routes.

Context

To set the convergence priority of OSPFv3 routes based on a specified IPv6 prefix list takes effect on the public network only.

OSPFv3 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.

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. 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 ipv6-prefix** *ipv6-prefix-name* [**index** *index-number*] *matchMode ipv6-address masklen* [**match-network**] [**greater-equal** *greater-equal-value*] [**less-equal** *less-equal-value*]

The IPv6 prefix list is configured.

Step 3 Run **ospfv3** [*process-id*]

The OSPFv3 process is started and 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

1.1.6.5.4 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

1.1.6.5.5 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

1.1.6.6 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**.

1.1.6.6.1 Configuring OSPFv3 to Import External Routes

Importing the routes discovered by other routing protocols can enrich OSPFv3 routing information.

Context

OSPFv3 is a link state-based routing protocol and cannot filter LSAs to be advertised. Therefore, OSPFv3 must filter routes when importing them so that only the filtered routes can be advertised.

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** { **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.

External routes are imported.

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

1.1.6.6.2 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-val*] *

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

1.1.6.6.3 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

1.1.6.6.4 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.

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 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 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, OSPFv3 filters all imported routes.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.6.6.5 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

1.1.6.6.6 (Optional) Configuring OSPFv3 to Discard Specified LSAs

Context

OSPFv3 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 **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

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

1.1.6.6.7 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 perform route summarization for intra-area routes.

     Route summarization is configured on an ABR.

     - The **not-advertise** parameter prevents the routes in the specified network segment from being advertised.
     - The **cost** *cost* parameter sets a cost for a 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 perform route summarization for the routes imported by OSPFv3.

     - The **cost** *cost* parameter sets a cost for a summary route.
     - The **tag** *tag* parameter sets a tag to control summary route advertisement.
     - The **not-advertise** parameter prevents the summary route with the specified IPv6 prefix or prefix length from being advertised.
     - The **distribute-delay** *interval* parameter sets a delay in advertising summary routes, which ensures that the summary routes advertised each time contain more valid routes and prevents network flapping caused by incorrect routing information.
  d. Run **commit**

The configuration is committed.

----End

1.1.6.6.8 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

1.1.6.7 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 1-55 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 checking in OSPFv3 LSAs on both the ASBRs, allowing them to use the received LSAs to calculate routes as normal.

Figure 1-55 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*] *

OSPFv3 is enabled to import BGP routes.

Step 5 Run **dn-bit-set disable** { **summary** | **ase** | **nssa** }

The OSPFv3 device is disabled from setting the DN bit in LSAs.

Step 6 Run **commit**

The configuration is committed.

----End

1.1.6.8 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, 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.

NOTE

The following steps are optional, and choose them as required.

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.

To disable OSPFv3 neighbor relationship flapping suppression globally, run the **suppress-flapping peer disable** command.

Step 2 Run **interface** *interface-type interface-number*

The interface view is displayed.

To disable OSPFv3 neighbor relationship flapping suppression from one of the interfaces, run the **ospfv3 suppress-flapping peer disable** command.

Step 3 Run **ospfv3 suppress-flapping peer hold-down** *interval* [**instance** *instance-id*]

The Hold-down mode is configured, and its duration is set.

Flapping suppression works in either of the following modes:

- 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: 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.

  NOTE

  To change the maximum cost of each OSPFv3 link, run the **maximum-link-cost** *cost* command.

Flapping suppression can also work first in Hold-down mode and then in Hold-max-cost mode.

To disable the Hold-max-cost mode, run the **ospfv3 suppress-flapping peer hold-max-cost disable** [**instance** *instance-id*] command.

Step 4 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*.

Using the default detection parameters is recommended.

Step 5 Run **quit**

Return to the system view.

Step 6 Run **quit**

Return to the user view.

Step 7 Run **reset ospfv3** *process-id* **suppress-flapping peer** [*interface-type interface-number*] [**notify-peer**]

Interfaces are forced to exit from OSPFv3 neighbor relationship flapping suppression.

NOTE

Interfaces exit from flapping suppression in the following scenarios:

- The suppression timer expires.
- The corresponding OSPFv3 process is reset.
- An OSPFv3 neighbor 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.

Step 8 Run **commit**

The configuration is committed.

----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.

1.1.6.9 Configuring Routing Loop Detection for Routes Imported to OSPFv3

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

If the device detects an OSPFv3 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 OSPFv3 routing loop alarm.

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

1.1.6.10 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.

1.1.6.11 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.

To disable OSPFv3 flush source tracing globally, run the **ospfv3 flush-source-trace disable** command.

Step 2 Run **interface** *interface-type interface-number*

The interface view is displayed.

To disable OSPFv3 flush source tracing on an interface, run the **ospfv3 flush-source-trace block** command.

Step 3 Run **quit**

Return to the system view.

Step 4 Run **ospfv3** [*process-id*] [**vpn-instance** *vpn-instance-name*]

The OSPFv3 process is started, and the OSPFv3 view is displayed.

Step 5 Run **quit**

Return to the system view.

Step 6 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, you can run the **ospfv3 flush-source-trace port** command to change the UDP port number.

Step 7 Run **quit**

Return to the user view.

Step 8 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.

Step 9 Run **commit**

The configuration is committed.

----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.

1.1.6.12 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

- Configure a dynamic OSPFv3 hostname.
  a. Run **system-view**

     The system view is displayed.
  b. Run **ospfv3** [*process-id*]

     The OSPFv3 process view is displayed.
  c. Run **hostname** *host-name*

     An OSPFv3 dynamic hostname is configured.

NOTE

If you specify *host-name* in the **hostname** command, *host-name* is advertised as the dynamic hostname. If no *host-name* is specified in the **hostname** command, the hostname specified in the **sysname** command is advertised as the dynamic hostname.

d. 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**

1.1.6.13 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 process 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 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.

1.1.6.14 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 **1.1.6.13 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 process 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.
- For Type 5 LSAs converted from Type 7 LSAs, to disable such LSAs from carrying forwarding address, 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.

1.1.6.15 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 1-56**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-56 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**.

1.1.6.15.1 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.

NOTE

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 1-57**, 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 1-57 Solution of selecting a backup path for OSPFv3 IP FRR

Step 7 Run **commit**

The configuration is committed.

----End

1.1.6.15.2 (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.

 Copyright © Huawei Technologies Co., Ltd.

1.1.6.15.3 (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

1.1.6.15.4 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

1.1.6.16 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**.

1.1.6.16.1 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

1.1.6.16.2 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* } *

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 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

1.1.6.16.3 (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.

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

1.1.6.16.4 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

1.1.6.17 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.

1.1.6.17.1 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.

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.

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.

  NOTE

  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.

 Copyright © Huawei Technologies Co., Ltd.

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

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 OSPFv3 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

1.1.6.17.2 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

1.1.6.17.3 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

1.1.6.17.4 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

1.1.6.17.5 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** *instanceId*] 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** *instanceId*] 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.

OSPFv3 authentication takes effect in descending order of priority as follows: interface authentication, area authentication, and process authentication.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.6.17.6 Verifying the Configuration of OSPFv3 Authentication

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.

----End

1.1.6.18 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**.

1.1.6.18.1 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.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.6.18.2 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

1.1.6.18.3 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 process view is displayed.

Step 3 Run **sham-hello enable**

OSPFv3 sham hello is enabled.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.6.18.4 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

On a stable network that requires fast route convergence, **intelligent-timer** needs to be specified in the **lsa-originate-interval** command to enable an intelligent timer to update OSPFv3 LSAs so that topology changes or route changes can be immediately advertised to the network through LSAs, which speeding up route convergence. This prevents too many network bandwidth and device resources from being consumed due to network connection changes or frequent route flapping.

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 the OSPFv3 LSAs to be sent flap is configured.

If no flapping occurs among the OSPFv3 LSAs to be sent, the configuration of the **lsa-originate-interval** command prevents the device from frequently sending LSAs. If the OSPFv3 LSAs to be sent 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

1.1.6.18.5 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

1.1.6.18.6 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

1.1.6.18.7 Verifying the Configuration of OSPFv3 Fast Convergence

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 in any view to view information about an OSPFv3 interface.
- Run the **display ospfv3** [*process-id*] command to check brief OSPFv3 information.

----End

1.1.6.19 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 NE40E 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.

1.1.6.20 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

1.1.6.21 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**.

1.1.6.21.1 Setting the Priority of OSPFv3

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* } *

The priority of OSPF is set.

- The parameter **ase** sets the priority for AS-external routes.
- The parameter **preference** specifies the priority for OSPF routes. The smaller the value, the higher the priority.
- **route-policy** *route-policy-name* specifies the name of a route-policy used to filter routes.
- **route-filter** *route-filter-name* specifies the name of a route-filter used to filter routes.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.6.21.2 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

1.1.6.21.3 Setting the Interval at Which LSAs Are Retransmitted Between Adjacent Routers

Control packet retransmission and improve the convergence rate by setting an interval at which an LSA packet is retransmitted to the adjacent Router.

Context

After sending an LSA packet to the adjacent Router, the Router 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*

 Copyright © Huawei Technologies Co., Ltd.

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 2^2

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 Router is configured.

NOTE

Set the LSA retransmission interval to a proper value on the Router. 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 Routers.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.6.21.4 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

1.1.6.21.5 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 Router. 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.
  a. Run **system-view**

     The system view is displayed.
  b. Run **ospfv3** [*process-id*]

     The OSPFv3 view is displayed.
  c. Run **spf timers** *delay-interval hold-interval*

     An SPF normal timer is configured.
  d. Run **commit**

     The configuration is committed.
- Configure an SPF intelligent timer.
  a. 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 ≥ 2) time equals *hold-interval-1* x $2^{(n-2)}$.

iii. After the interval specified by *hold-interval-1* x $2^{(n-2)}$ 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

1.1.6.21.6 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 Router 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

1.1.6.21.7 Verifying the Configuration of OSPFv3 Network Stability

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 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*] **statistics maxage-lsa** command to check information about MaxAge Router-LSAs.

----End

1.1.6.21.8 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.

1.1.6.22 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.

1.1.6.23 Configuring Micro-Isolation CAR for OSPFv3

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 packets of non-OSPFv3 neighbors 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

1.1.6.24 Configuring the Threshold Alarm Function for the Number of OSPFv3 Neighbors

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

1.1.6.25 Maintaining OSPFv3

Maintaining OSPFv3 involves resetting OSPFv3 and clearing OSPFv3 statistics.

1.1.6.25.1 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

1.1.6.25.2 Resetting OSPFv3

Restarting OSPFv3 can reset OSPFv3.

Context

NOTICE

The OSPF neighbor relationships will be torn down if you run the **reset ospfv3** command to reset OSPF connections. Exercise caution when running the command.

After modifying the OSPFv3 routing policy or protocol, reset the OSPFv3 connection to validate the modification. To reset OSPFv3 connections, run the **reset ospfv3** command 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 reset OSPFv3 route redistribution.

----End

1.1.6.25.3 Disabling OSPFv3 Memory Overload Control

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.

NOTE

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

1.1.6.25.4 Disabling OSPFv3 CPU Overload Control

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.

NOTE

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

1.1.6.26 Configuration Examples for OSPFv3

This section provides several OSPFv3 configuration examples.

1.1.6.26.1 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 1-58**, 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 1-58 Networking for configuring OSPFv3 areas

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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: 2001:DB8:4::1/64 and 2001:DB8:2::2/64
Device B	2.2.2.2	1	Area 0: 2001:DB8:1::1/64 Area 1: 2001:DB8:2::1/64
Device C	3.3.3.3	1	Area 0: 2001:DB8:1::2/64 Area 2: 2001:DB8:3::1/64
Device D	4.4.4.4	1	Area 2: 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 gigabitethernet0/3/0
[*DeviceA-GigabitEthernet0/3/0] ospfv3 1 area 1
[*DeviceA-GigabitEthernet0/3/0] quit
[*DeviceA] interface gigabitethernet0/2/0
[*DeviceA-GigabitEthernet0/2/0] ospfv3 1 area 1
[*DeviceA-GigabitEthernet0/2/0] 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 gigabitethernet0/1/0
[*DeviceB-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet0/2/0
[*DeviceB-GigabitEthernet0/2/0] ospfv3 1 area 1
[*DeviceB-GigabitEthernet0/2/0] quit
[*DeviceB] commit
```

Configure DeviceC.

```
[~DeviceC] ospfv3
[*DeviceC-ospfv3-1] router-id 3.3.3.3
[*DeviceB-ospfv3-1] area 0.0.0.0
[*DeviceB-ospfv3-1] area 0.0.0.2
[*DeviceC-ospfv3-1] quit
[*DeviceC] interface gigabitethernet 0/1/0
[*DeviceC-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ospfv3 1 area 2
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] commit
```

Configure DeviceD.

```
[~DeviceD] ospfv3
[*DeviceD-ospfv3-1] router-id 4.4.4.4
[*DeviceB-ospfv3-1] area 0.0.0.2
[*DeviceD-ospfv3-1] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] ospfv3 1 area 2
[*DeviceD-GigabitEthernet0/2/0] 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)


  OSPFv3 Area (0.0.0.1)
  Neighbor ID    Pri   State       Dead Time   Interface              Instance ID
  1.1.1.1        1     Full/ -     00:00:34    GigabitEthernet0/2/0   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    GigabitEthernet0/1/0   0
```

Display the OSPFv3 neighbors of DeviceC.

```
[*DeviceC] display ospfv3 peer
```

```
 OSPFv3 Process (1)


 OSPFv3 Area (0.0.0.0)
 Neighbor ID     Pri   State       Dead Time   Interface                 Instance ID
 2.2.2.2         1     Full/ -     00:00:37    GigabitEthernet0/1/0      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    GigabitEthernet0/2/0      0
```

Display the OSPFv3 routing table of DeviceD.

```
[*DeviceD] display ospfv3 routing
OSPFv3 Process (1)

  Destination                                  Metric
    Next-hop
 IA 2001:DB8:1::/64                                 2
         via FE80::1572:0:5EF4:1, GigabitEthernet0/2/0
 IA 2001:DB8:2::/64                                 3
         via FE80::1572:0:5EF4:1, GigabitEthernet0/2/0
    2001:DB8:3::/64                                 1
         directly-connected, GigabitEthernet0/2/0
 IA 2001:DB8:4::/64                                 4
         via FE80::1572:0:5EF4:1, GigabitEthernet0/2/0
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
 ospfv3 1 area 0.0.0.1
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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
```

1.1.6.26.2 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 1-59**, 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 1-59 Networking for configuring OSPFv3 DR election

NOTE

Interface 1 in this example represents GE 0/1/0.

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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceB-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceC-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceD-GigabitEthernet0/1/0] 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)


  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    GE0/1/0        0
  3.3.3.3         1   Full/Backup      00:00:36    GE0/1/0        0
  4.4.4.4         1   Full/DR          00:00:38    GE0/1/0        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)
```

```
 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    GE0/1/0         0
 2.2.2.2         1   Full/DROther     00:00:35    GE0/1/0         0
 3.3.3.3         1   Full/Backup      00:00:30    GE0/1/0         0
```

Step 3 Set the DR priority of the interface.

Set the DR priority of DeviceA to 100.

```
[~DeviceA] interface gigabitethernet 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ospfv3 dr-priority 100
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] commit
```

Set the DR priority of DeviceB to 0.

```
[~DeviceB] interface gigabitethernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] ospfv3 dr-priority 0
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] commit
```

Set the DR priority of DeviceC to 2.

```
[~DeviceC] interface gigabitethernet 0/1/0
[~DeviceC-GigabitEthernet0/1/0] ospfv3 dr-priority 2
[*DeviceC-GigabitEthernet0/1/0] 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)


 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    GE0/1/0         0
 3.3.3.3         2   Full/Backup      00:00:38    GE0/1/0         0
 4.4.4.4         1   Full/DR          00:00:31    GE0/1/0         0
```

Display the neighbors of DeviceD. The command output shows that DeviceD remains the DR.

```
[~DeviceD] display ospfv3 peer
 OSPFv3 Process (1)


 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    GE0/1/0         0
 2.2.2.2         0   Full/DROther       00:00:30    GE0/1/0         0
 3.3.3.3         2   Full/Backup        00:00:36    GE0/1/0         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)


  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    GE0/1/0        0
  3.3.3.3         2   Full/Backup      00:00:36    GE0/1/0        0
  4.4.4.4         1   Full/DROther     00:00:39    GE0/1/0        0
```

Check the neighbors of DeviceD. The command output shows that DeviceA functions as the DR.

```
[~DeviceD] display ospfv3 peer
  OSPFv3 Process (1)


  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    GE0/1/0        0
  2.2.2.2         0   2-Way/DROther    00:00:35    GE0/1/0        0
  3.3.3.3         2   Full/Backup      00:00:39    GE0/1/0        0
```

----End

Configuration Files

- DeviceA configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

1.1.6.26.3 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 1-60**:

- 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 1-60 Networking for configuring OSPFv3 IP FRR

NOTE

Interfaces 1 through 4 in this example represent GE0/1/0, GE0/2/0, GE0/3/0, and GE0/1/1, respectively.

Context

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.

During the configuration of OSPFv3 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 associated with the link status of an interface so that link faults can be rapidly detected.

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. (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	GE0/1/0	2001:DB8:1000::1/96
		GE0/2/0	2001:DB8:1001::1/96
		GE0/3/0	2001:DB8:1002::1/96
Device A	2.2.2.2	GE0/1/0	2001:DB8:1000::2/96
		GE0/2/0	2001:DB8:2000::2/96
Device N	3.3.3.3	GE0/1/0	2001:DB8:1002::2/96
		GE0/2/0	2001:DB8:2002::2/96
Device E	4.4.4.4	GE0/1/0	2001:DB8:2000::1/96
		GE0/2/0	2001:DB8:2001::1/96
		GE0/3/0	2001:DB8:2002::1/96
		GE0/1/1	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 gigabitethernet0/1/0
[~DeviceS-GigabitEthernet0/1/0] ospfv3 cost 5
[*DeviceS-GigabitEthernet0/1/0] quit
```

```
[*DeviceS] interface gigabitethernet0/2/0
[*DeviceS-GigabitEthernet0/2/0] ospfv3 cost 15
[*DeviceS-GigabitEthernet0/2/0] quit
[*DeviceS] interface gigabitethernet0/3/0
[*DeviceS-GigabitEthernet0/3/0] ospfv3 cost 10
[*DeviceS-GigabitEthernet0/3/0] quit
[*DeviceS] commit
```

Configure Device A.

```
[~DeviceA] interface gigabitethernet0/1/0
[~DeviceA-GigabitEthernet0/1/0] ospfv3 cost 5
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet0/2/0
[*DeviceA-GigabitEthernet0/2/0] ospfv3 cost 5
[*DeviceA-GigabitEthernet0/2/0] quit
[*DeviceA] commit
```

Configure Device N.

```
[~DeviceN] interface gigabitethernet0/1/0
[~DeviceN-GigabitEthernet0/1/0] ospfv3 cost 10
[*DeviceN-GigabitEthernet0/1/0] quit
[*DeviceN] interface gigabitethernet0/2/0
[*DeviceN-GigabitEthernet0/2/0] ospfv3 cost 10
[*DeviceN-GigabitEthernet0/2/0] quit
[*DeviceN] commit
```

Step 5 Block FRR on the specified interface on Device S.

```
[~DeviceS] interface gigabitethernet0/1/0
[~DeviceS-GigabitEthernet0/1/0] ospfv3 frr block
[*DeviceS-GigabitEthernet0/1/0] 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] 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
 OSPFv3 Process (1)
   Destination                          Metric
     Nexthop
   2001:DB8:2000:1::/64                    3124
       via 2001:DB8:2001::1/96, GE0/2/0
         backup via FE80::2000:10FF:4, GE0/3/0, LFA LINK-NODE
     Priority      :Low
```

The command output shows that a backup link is generated by FRR calculation on Device S.

----End

Configuration Files

- Device S configuration file

```
#
sysname DeviceS
#
 bfd
```

```
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1000::1/96
 ospfv3 cost 5
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1001::1/96
 ospfv3 cost 15
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1000::2/96
 ospfv3 cost 5
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1002::2/96
 ospfv3 cost 10
#
```

```
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2000::1/96
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2001::1/96
#
interface GigabitEthernet0/3/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2002::1/96
#
interface GigabitEthernet0/1/1
 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
```

1.1.6.26.4 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 1-61**, 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 1-61 Networking for configuring BFD for OSPFv3

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/1/1, and GE 0/1/2, 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/1/1
[*DeviceA-GigabitEthernet0/1/1] ospfv3 1 area 0.0.0.0
[*DeviceA-GigabitEthernet0/1/1] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet 0/1/1
[*DeviceB-GigabitEthernet0/1/1] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/1/1] quit
[*DeviceB] interface gigabitethernet 0/1/2
[*DeviceB-GigabitEthernet0/1/2] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/1/2] commit
```

Configure Device C.

```
[~DeviceC] ospfv3 1
[*DeviceC-ospfv3-1] router-id 3.3.3.3
[*DeviceC-ospfv3-1] quit
[*DeviceC] interface gigabitethernet 0/1/0
[*DeviceC-GigabitEthernet0/1/0] ospfv3 1 area 0.0.0.0
[DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/1/1
[*DeviceC-GigabitEthernet0/1/1] ospfv3 1 area 0.0.0.0
[*DeviceC-GigabitEthernet0/1/1] 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)
 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   GE0/1/0             0
3.3.3.3          1 Full/-           00:00:37   GE0/1/1             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
      Next-hop
    2001:DB8:1::/64                                        1
      directly connected, GE0/1/0, Flags : A
    2001:DB8:2::/64                                        2
      via FE80::3AE9:7BFF:FE31:307, GE0/1/1, Flags : A
      via FE80::3AE9:7BFF:FE21:300, GE0/1/0, Flags : A
    2001:DB8:3::/64                                        1
      directly connected, GE0/1/1, Flags : A
    2001:DB8:4::1/64                                        1
      via FE80::3AE9:7BFF:FE21:300, GE0/1/0
```

As shown in the OSPFv3 routing table, the next hop address of the route to 2001:DB8:4::1/64 is GigabitEthernet 0/1/0, 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 Device B as an example.

```
<DeviceB> display ospfv3 bfd session
* - STALE

OSPFv3 Process (1)

 Neighbor-Id     Interface                    BFD-Status
 1.1.1.1         GE0/1/0                      Up
 3.3.3.3         GE0/1/1                      Up
```

Step 4 Verify the configuration.

Run the **shutdown** command on GE 0/1/0 of Device B to simulate a primary link failure.

```
[~DeviceB] interface gigabitethernet0/1/0
[~DeviceB-GigabitEthernet0/1/0] shutdown
[*DeviceB-GigabitEthernet0/1/0] 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 0/1/1. 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, GE0/1/1, Flags : A
    2001:DB8:3::/64                                         1
      directly connected, GE0/1/1, Flags : A
    2001:DB8:4::/64                                         3
      via FE80::3AE9:7BFF:FE31:307, GE0/1/1, 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 gigabitethernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::3/64
 ospfv3 1 area 0.0.0.0
#
interface gigabitethernet0/1/1
 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 gigabitethernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 ospfv3 1 area 0.0.0.0
#
interface gigabitethernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/64
 ospfv3 1 area 0.0.0.0
#
interface gigabitethernet0/1/2
 undo shutdown
```

 Copyright © Huawei Technologies Co., Ltd.

```
 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 gigabitethernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
 ospfv3 1 area 0.0.0.0
#
interface gigabitethernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::3/64
 ospfv3 1 area 0.0.0.0
#
return
```

1.1.6.26.5 Example for Configuring IPSec for OSPFv3

Networking Requirements

As shown in **Figure 1-62**, Device A and Device B run Open Shortest Path First version 3 (OSPFv3) and are reachable.

Figure 1-62 IPsec for OSPFv3

NOTE

Interface 1 in this example represents GE 0/1/1.

If no authentication mechanism is configured, IP protocol packets along the route between DeviceA and DeviceB may be modified or faked, causing neighbor relationships between DeviceA and DeviceB to be interrupted or incorrect routes to be imported.

To prevent such attacks, IPsec can be configured between DeviceA and DeviceB to protect OSPFv3 protocol packets during transmission. ESP is configured as the security protocol, and SHA2-256 is configured as the authentication algorithm.

Precautions

- The encapsulation modes and security protocols on both IPsec peers must be identical.
- The authentication modes and encryption algorithms on both IPsec peers must be identical.
- The SPIs and authentication keys on both IPsec peers must be identical.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPFv3 functions on Device A and Device B.
2. Configure an IPsec proposal, set the security protocol used for the proposal to ESP, and specify the authentication algorithm SHA2-256 and the encryption algorithm AES-GCM-128 for ESP.
3. Configure an SA and apply a proposal to SA, define inbound and outbound parameters which include SPI and keys.
4. Apply the SA to the OSPFv3 process to protect OSPFv3 protocol packets 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 the String Format
Device A	1.1.1.1	1	12345	abcdef
Device B	2.2.2.2	1	12345	abcdef

Procedure

Step 1 Configure OSPFv3 on Device A and Device B.

Configure Device A.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceA
[*HUAWEI] commit
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] router-id 1.1.1.1
[*DeviceA-ospfv3-1] area 1
[*DeviceA-ospfv3-1-area-0.0.0.1] commit
[~DeviceA-ospfv3-1-area-0.0.0.1] quit
```

Configure Device B.

```
<HUAWEI> system-view
[~HUAWEI] sysname DeviceB
[*HUAWEI] commit
[~DeviceB] ospfv3 1
[*DeviceB-ospfv3-1] router-id 2.2.2.2
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*DeviceB-ospfv3-1] area 1
[*DeviceB-ospfv3-1-area-0.0.0.1] commit
[~DeviceB-ospfv3-1-area-0.0.0.1] quit
```

Step 2 Configure IPv6 addresses and enable OSPFv3 on interfaces.

Configure Device A.

```
[~DeviceA] interface gigabitethernet0/1/1
[~DeviceA-GigabitEthernet0/1/1] ipv6 enable
[*DeviceA-GigabitEthernet0/1/1] ipv6 address 2001:db8::1 64
[*DeviceA-GigabitEthernet0/1/1] ospfv3 1 area 1
[*DeviceA-GigabitEthernet0/1/1] commit
[~DeviceA-GigabitEthernet0/1/1] quit
```

Configure Device B.

```
[~DeviceB] interface gigabitethernet0/1/1
[~DeviceB-GigabitEthernet0/1/1] ipv6 enable
[*DeviceB-GigabitEthernet0/1/1] ipv6 address 2001:db8::2 64
[*DeviceB-GigabitEthernet0/1/1] ospfv3 1 area 1
[*DeviceB-GigabitEthernet0/1/1] commit
[~DeviceB-GigabitEthernet0/1/1] quit
```

Step 3 Configure security proposals on Device A and Device B.

Configure a security proposal on Device A.

```
[~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-gcm-128
[*DeviceA-ipsec-proposal-proposal1] esp authentication-algorithm sha2-256
[*DeviceA-ipsec-proposal-proposal1] commit
[~DeviceA-ipsec-proposal-proposal1] quit
```

Configure a security proposal on Device B.

```
[~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-gcm-128
[*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 Device A and Device B to view configurations. Use the display on Device A as an example.

```
[~DeviceA] display ipsec proposal
Total IP security proposal number: 1
IP security proposal name: proposal1
encapsulation mode: transport
transform: esp-new
ESP protocol: authentication SHA2-HMAC-256, not use encryption
```

Step 4 Configure SAs and apply them to Device A and Device B.

Configure an SA and apply it to Device A.

```
[~DeviceA] ipsec sa sa1
[*DeviceA-ipsec-sa-sa1] proposal proposal1
[*DeviceA-ipsec-sa-sa1] commit
```

Configure an SA and apply it to Device B.

```
[~DeviceB] ipsec sa sa2
[*DeviceB-ipsec-sa-sa2] proposal proposal2
[*DeviceB-ipsec-sa-sa2] commit
```

Step 5 Configure SPIs and authentication keys in the string format on Device A and Device B.

Configure SPIs and authentication keys in the string format on Device A.

```
[~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 the string format on Device B.

```
[~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 Configure SAs for OSPFv3 processes.

Configure an SA for the OSPFv3 process on Device A.

```
[~DeviceA] ospfv3 1
[*DeviceA-ospfv3-1] ipsec sa sa1
[*DeviceA-ospfv3-1] commit
```

Configure an SA for the 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 Device A and Device B to view configurations. Use the display on Device A as an example.

```
[~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 view 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
      cannot 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

- Configuration file of DeviceA

```
#
sysname DeviceA
#
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\{#%#%#
#
ospfv3 1
 router-id 1.1.1.1
 ipsec sa sa1
 area 0.0.0.1
#
interface GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8::1/64
 ospfv3 1 area 0.0.0.1
```

```
#
return
```

- Configuration file of DeviceB

```
#
sysname DeviceB
#
ipsec proposal proposal2
 encapsulation-mode transport
 esp authentication-algorithm sha2-256
 esp encryption-algorithm aes-gcm-128
#
ipsec sa sa2
 proposal proposal2
 sa spi inbound esp 12345
 sa string-key inbound esp %#%#<}j/@XSE9JZT5]2"T#]2"T<}j/@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 0.0.0.1
#
interface GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8::2/64
 ospfv3 1 area 0.0.0.1
#
return
```

1.1.6.26.6 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 1-63**, 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. To reduce the routing table size, simplify route management, and improve network stability, configure the ASBR to summarize the three static routes into route 2001::/16 and advertise the summary route only to area 0.

Figure 1-63 Example for configuring OSPFv3 route summarization on an ASBR

NOTE

Interface 1 in this example represents GE 0/1/0.

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 gigabitethernet0/1/0
[~DeviceA-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceA-GigabitEthernet0/1/0] 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 gigabitethernet0/1/0
[~ASBR-GigabitEthernet0/1/0] ospfv3 1 area 0
[*ASBR-GigabitEthernet0/1/0] 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)
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   GE0/1/0              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::1/128, 2001:DB8:3::1/128, and 2001:DB8:4::1/128, indicating that route summarization is not performed.

```
[~ASBR] display ospfv3 lsdb
         OSPFv3 Router with ID (2.2.2.2) (Process 1)

             Link-LSA (Interface GE0/1/0)
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::/16.

```
[~ASBR] ospfv3 1
[*ASBR-ospfv3-1] asbr-summary 2001:: 16
[*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::/16.

```
[~ASBR] display ospfv3 lsdb

         OSPFv3 Router with ID (2.2.2.2) (Process 1)

             Link-LSA (Interface GE0/1/0)
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::/16
    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::                     16          3 [Active]     Advertised
```

----End

Configuration Files

- Device A configuration file

```
#
sysname Device A
#
ospfv3 1
 router-id 1.1.1.1
 area 0.0.0.0
#
interface GigabitEthernet0/1/0
 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:: 16
 area 0.0.0.0
#
interface GigabitEthernet0/1/0
 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
```

1.1.6.26.7 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 1-64**, 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 1-64 Example for configuring OSPFv3 route summarization on an ABR

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 gigabitethernet0/2/0
[~DeviceA-GigabitEthernet0/2/0] ospfv3 1 area 1
[*DeviceA-GigabitEthernet0/2/0] 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 gigabitethernet0/1/0
[~DeviceB-GigabitEthernet0/1/0] ospfv3 1 area 0
[*DeviceB-GigabitEthernet0/1/0] 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 gigabitethernet0/3/0
[~DeviceC-GigabitEthernet0/3/0] ospfv3 1 area 1
[*DeviceC-GigabitEthernet0/3/0] 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 gigabitethernet0/1/0
[~ABR-GigabitEthernet0/1/0] ospfv3 1 area 0
[*ABR-GigabitEthernet0/1/0] quit
[*ABR] interface gigabitethernet0/2/0
[*ABR-GigabitEthernet0/2/0] ospfv3 1 area 1
[*ABR-GigabitEthernet0/2/0] quit
[*ABR] interface gigabitethernet0/3/0
[*ABR-GigabitEthernet0/3/0] ospfv3 1 area 1
[*ABR-GigabitEthernet0/3/0] 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)
```

```
OSPFv3 Area (0.0.0.0)
Neighbor ID      Pri State           Dead Time  Interface          Instance ID
1.1.1.1            1 Full/DR          00:00:35   GE0/1/0                0
OSPFv3 Area (0.0.0.1)
Neighbor ID      Pri State           Dead Time  Interface          Instance ID
4.4.4.4            1 Full/Backup      00:00:40   GE0/3/0                0
3.3.3.3            1 Full/Backup      00:00:31   GE0/2/0                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 GE0/1/0)
Link State ID   Origin Router    Age   Seq#       CkSum      Prefix
0.0.0.18        1.1.1.1          1740  0x80000010 0x8b9b        1
0.0.0.18        2.2.2.2          1684  0x80000010 0xc249        1

             Link-LSA (Interface GE0/3/0)
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 GE0/2/0)
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 GE0/1/0)
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 GE0/3/0)
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 GE0/2/0)
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 Device A
#
ospfv3 1
 router-id 3.3.3.3
 area 0.0.0.1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:2::1/64
 ospfv3 1 area 0.0.0.1
#
return
```

- Device B configuration file

```
#
sysname Device B
#
ospfv3 1
 router-id 1.1.1.1
 area 0.0.0.0
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:1::2/64
 ospfv3 1 area 0.0.0.0
#
return
```

- Device C configuration file

```
#
sysname Device C
#
ospfv3 1
 router-id 4.4.4.4
 area 0.0.0.1
#
```

```
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:1::1/64
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:2::2/64
 ospfv3 1 area 0.0.0.1
#
interface GigabitEthernet0/3/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:3::2/64
 ospfv3 1 area 0.0.0.1
#
return
```

1.1.6.26.8 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 AS, without learning external routes from other areas in the AS, which reduces bandwidth and storage resource consumption on the Router.

In **Figure 1-65**, 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 1-65 Networking for configuring an OSPFv3 NSSA

NOTE

Interfaces 1 through 3 in this example represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 1-65**. For details about the configuration, see Configuration Files in this section.

Step 2 Configure basic OSPFv3 functions.

For detailed operations, see **1.1.6.26.1 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
```

```
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

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 GigabitEthernet0/1/0
 undo shutdown
 port default vlan 18
#
interface GigabitEthernet0/2/0
 undo shutdown
 port default vlan 13
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 port default vlan 16
#
interface GigabitEthernet0/2/0
 undo shutdown
 port default vlan 14
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 port default vlan 18
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 port default vlan 13
#
interface GigabitEthernet0/2/0
 undo shutdown
 port default vlan 11
#
ipv6 route-static 2001:DB8:6::1 128 NULL0
#
return
```

1.1.6.26.9 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 1-66**, 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 1-66 Routing loop detection for OSPFv3 multi-process mutual route import

NOTE

Interfaces 1, 2, and 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 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 Configure an IP address for each interface.

DeviceA is used as an example.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ipv6 enable
[*DeviceA-GigabitEthernet0/1/0] ipv6 address 2001:db8:1::1 64
[*DeviceA-GigabitEthernet0/1/0] ipv6 address FE80::A:B link-local
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/2/0
[*DeviceA-GigabitEthernet0/2/0] ipv6 enable
[*DeviceA-GigabitEthernet0/2/0] ipv6 address 2001:db8:2::2 64
[*DeviceA-GigabitEthernet0/2/0] ipv6 address FE80::A:F link-local
[*DeviceA-GigabitEthernet0/2/0] 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 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] interface gigabitethernet 0/3/0
[*DeviceC-GigabitEthernet0/3/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet0/3/0] 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-0.0.0.0] quit
[*DeviceB-ospfv3-1] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/2/0] quit
[*DeviceB] interface gigabitethernet 0/3/0
[*DeviceB-GigabitEthernet0/3/0] ospfv3 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/3/0] 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-0.0.0.0] quit
[*DeviceA-ospfv3-2] quit
[*DeviceA] interface gigabitethernet 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ospfv3 2 area 0.0.0.0
```

```
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/2/0
[*DeviceA-GigabitEthernet0/2/0] ospfv3 2 area 0.0.0.0
[*DeviceA-GigabitEthernet0/2/0] 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 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 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 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)
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   GE0/2/0           0
3.3.3.1          1 Full/DR          00:00:36   GE0/3/0           0

OSPFv3 Process (2)
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   GE0/1/0           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    : GE0/2/0                  Flags        : D

Destination  : 2001:db8:44::44          PrefixLength : 64
NextHop      : FE80::C:B                Preference   : 150
Cost         : 1                        Protocol     : OSPFv3ASE
RelayNextHop : ::                       TunnelID     : 0x0
Interface    : GE0/3/0                  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    : GE0/1/0                  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    : GE0/3/0                  Flags        : D

Destination  : 2001:db8:44::44          PrefixLength : 64
NextHop      : FE80::A:F                Preference   : 150
Cost         : 1                        Protocol     : OSPFv3ASE
RelayNextHop : ::                       TunnelID     : 0x0
Interface    : GE0/1/0                  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    : GE0/3/0                        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    : GE0/1/0                        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    : GE0/3/0                        Flags        : D
```

The preceding command output shows 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 0.0.0.0
#
route loop-detect ospfv3 enable
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 0.0.0.0
#
ospfv3 2
 router-id 2.2.2.2
 import-route ospfv3 1
 area 0.0.0.0
#
route loop-detect ospfv3 enable
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 ipv6 address FE80::B:A link-local
 ospfv3 2 area 0.0.0.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:5::2/64
 ipv6 address FE80::B:F link-local
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::2/64
 ipv6 address FE80::B:C link-local
 ospfv3 1 area 0.0.0.0
#
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 0.0.0.0
#
route loop-detect ospfv3 enable
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::1/64
 ipv6 address FE80::C:B link-local
 ospfv3 1 area 0.0.0.0
#
interface GigabitEthernet0/2/0
```

```
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:7::2/64
 ospfv3 1 area 0.0.0.0
 isis ipv6 enable 1
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:6::1/64
 isis ipv6 enable 1
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
#
route loop-detect ospfv3 enable
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:7::1/64
 ospfv3 1 area 0.0.0.0
 isis ipv6 enable 1
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/3/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:4::2/64
 ospfv3 1 area 0.0.0.0
#
return
```

1.1.7 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.

1.1.7.1 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.

1.1.7.2 Configuration Precautions for RIP

Feature Requirements

Table 1-13 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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

1.1.7.3 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.

1.1.7.3.1 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.

1.1.7.3.2 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.
  a. Run **system-view**

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

     A RIP process is created, and the view of the RIP process is displayed.
  c. Run **network** *network-address*

     RIP is enabled on a specified network segment.
  d. 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.
  a. Run **system-view**

     The system view is displayed.
  b. Run **interface** *interface-type interface-number*

     The interface view is displayed.
  c. Run **rip enable** *process-id*

     RIP is enabled on all the network segments where the interface is located.
  d. Run **commit**

     The configuration is committed.

----End

1.1.7.3.3 (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.
  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 **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

1.1.7.3.4 (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

1.1.7.3.5 (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

1.1.7.3.6 (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

1.1.7.3.7 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

1.1.7.3.8 Verifying the Basic RIP Configuration

After configuring basic RIP functions, verify the current running status of RIP and RIP routing information.

Prerequisites

Basic RIP functions have been configured.

Procedure

- Run the **display rip** [*process-id* | **vpn-instance** *vpn-instance-name*] command to check the running status and RIP configurations.
- Run the **display rip** *process-id* **route** command to check RIP routes.

----End

1.1.7.4 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.
- **1.1.7.3 Configuring Basic RIP Functions**

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.7.4.1 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.

1.1.7.4.2 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.

1.1.7.4.3 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 **1.1.7.7.2 Configuring RIP Timers**.

1.1.7.4.4 Verifying the Configuration of Routing Loop Prevention

After configuring routing loop prevention, verify the 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 the running status and configuration of RIP.
- 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

1.1.7.5 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.
- **1.1.7.3 Configuring Basic RIP Functions**.

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.7.5.1 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 1-67**, 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 1-67 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

1.1.7.5.2 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* }
  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 the IP prefix:

  Run **rip metricin** { *value* | **ip-prefix** *ip-prefix-name value1* }

Step 4 Set the metric to be added to routes to be sent.

Run any of the following commands as required:

- Based on the basic ACL:
  a. Run **rip metricout** { *value* | { *acl-number* | **acl-name** *acl-name* } *value1* } *
  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 basic 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*] *

     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 the IP prefix:

  Run **rip metricout** { *value* | **ip-prefix** *ip-prefix-name value1* } *

NOTE

When using an ACL or an IP prefix list with the **rip metricout** command, you can specify the metric to be added to the RIP route that matches the ACL or the IP prefix list. 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

1.1.7.5.3 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

1.1.7.5.4 Verifying the Configuration of RIP Route Selection

After adjusting RIP route selection, verify the running status of RIP, information about interfaces, and RIP routing information.

Prerequisites

RIP route selection has been adjusted.

Procedure

- Run the **display rip** [*process-id* | **vpn-instance** *vpn-instance-name*] command to check the running status and configuration of RIP.
- 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

1.1.7.6 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
- **1.1.7.3 Configuring Basic RIP Functions**

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.7.6.1 Configuring RIP to Import External Routes

RIP can import routes from other processes or routing protocols to enrich the RIP routing table.

Context

On a large-scale network, multiple routing protocols may be configured for devices in different areas. In this situation, configure RIP to import routes from other processes or routing protocols to implement inter-area communication.

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 you do not specify the cost for external routes, the value 0 takes effect.

Step 4 Import external routes.

- To import direct, static, IS-IS, or OSPF routes, or routes of other RIP processes, 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* }] * command or the **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

Import of routes of other protocols may lead to routing loops. Therefore, exercise caution when running the **import-route** command.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.7.6.2 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

1.1.7.6.3 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

1.1.7.6.4 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:

  a. Run **filter-policy** { *acl-number* | **acl-name** *acl-name* } **export** [**bgp** | **direct** | **static** | **unr** | { **isis** | **ospf** | **rip** } [*process-id*] | *interface-type interface-number*]

  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 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

1.1.7.6.5 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

1.1.7.6.6 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.

1.1.7.6.7 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 1-68**, 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 1-68 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

1.1.7.6.8 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 1-69**, 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 1-69 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

1.1.7.6.9 Verifying the Configuration of RIP Routing Information Control

After controlling RIP routing information, verify all activated routes in the RIP database and the running status of RIP.

Prerequisites

RIP routing information has been controlled.

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 configuration of the RIP process.

----End

1.1.7.7 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.
- **1.1.7.3 Configuring Basic RIP Functions**

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.7.7.1 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 the advertised summarized route, which prevents routing loops.

  d. Run **commit**

  The configuration is committed.

----End

1.1.7.7.2 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 **1.1.7.4.3 Configuring Suppression Timers**.

1.1.7.7.3 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

1.1.7.7.4 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

1.1.7.7.5 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

1.1.7.7.6 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

1.1.7.7.7 Verifying the Configuration of RIP Fast Convergence

After configuring RIP fast convergence, verify the running status of RIP, RIP routing information, all active routes in the RIP database, and information about interfaces.

Prerequisites

RIP fast convergence has been configured.

Procedure

- Run the **display rip** [*process-id* | **vpn-instance** *vpn-instance-name*] command to check the running status and configuration of RIP.
- 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 RIP routes in the database.
- Run the **display rip** *process-id* **interface** [*interface-type interface-number*] [**verbose**] command to check information about RIP interfaces.

----End

1.1.7.8 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.7.9 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 can rapidly detect faults on links and speed up response to network topology changes. You can configure static BFD on devices that require timely response to faults to implement basic BFD functions.

Static BFD must be enabled and session parameters must be set to implement static BFD for RIP.

Pre-configuration Tasks

Before configuring static BFD for RIP, complete the following tasks:

- Assign an IP address to each interface to ensure IP connectivity.
- **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.

1. Run **system-view**

   The system view is displayed.
2. Run **bfd**

   BFD is enabled globally.
3. Run **quit**

   Return to the system view.

Step 2 Configure static BFD for RIP.

1. Run **bfd** *session-name* **bind peer-ip** *peer-ip* [**interface** *interface-type interface-number*] [**source-ip** *source-ip*]

   BFD is bound to the interface.

   If both a peer IP address and a local interface are specified, BFD detects only a single-hop link (the route with the local interface as the outbound interface and with the peer IP address as the next hop address).

2. Set discriminators.
   - Run **discriminator local** *discr-value*

     The local discriminator is set.
   - Run **discriminator remote** *discr-value*

     The remote discriminator is set.

   The local discriminator must be the remote discriminator of the device on the other end; otherwise, a BFD session cannot be established.

   NOTE

   local *discr-value* set on the local device must be the same as **remote** *discr-value* set on the remote device. **remote** *discr-value* set on the local device must be the same as **local** *discr-value* set on the remote device.

3. Run **quit**

   Return to the system view.

Step 3 Enable static BFD on an interface.

1. Run **interface** *interface-type interface-number*

   The view of the specified interface is displayed.

2. Run **rip bfd static**

   Static BFD is enabled on the interface.

Step 4 Run **commit**

The configuration is committed.

----End

Checking the Configurations

After configuring static BFD for RIP, run the **display rip** *process-id* **interface** [*interface-type interface-number*] **verbose** command to check BFD for RIP configurations on the specified interface.

1.1.7.10 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 1-70 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 1-70 Networking diagram of unaffiliated static BFD for RIP

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.

1.1.7.11 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.
- **1.1.7.3 Configuring Basic RIP Functions**

Configuration Procedure

Perform one or more of the following configuration tasks (excluding verifying the configuration) as required.

1.1.7.11.1 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 1-71** shows a P2P topology, and an upgrade is performed. DeviceA and DeviceB use the default configuration.

Figure 1-71 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 1-72** shows a broadcast topology, and an upgrade from RIP-1 (or RIP-1-compatible version) to RIP-2 is performed on devices.

Figure 1-72 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 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 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.

1.1.7.11.2 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

1.1.7.11.3 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.

NOTE

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.

   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

1.1.7.11.4 Verifying the Configuration of RIP Network Security

After improving the security of a RIP network, verify information about RIP interfaces and the running status of RIP.

Prerequisites

Configurations have been made to improve security of a RIP network.

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 configuration of the RIP process.

----End

1.1.7.12 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.
- **1.1.7.3 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.

1.1.7.13 Maintaining RIP

RIP maintenance is implemented through debugging which affects system performance.

1.1.7.13.1 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

1.1.7.13.2 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

1.1.7.14 Configuration Examples for RIP

This section provides RIP configuration examples.

1.1.7.14.1 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 1-73**, 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 1-73 Configuring basic RIP functions

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 gigabitethernet0/1/0
[*DeviceA-GigabitEthernet0/1/0] rip authentication-mode hmac-sha256 cipher YsHsjx_202206 200
[*DeviceA-GigabitEthernet0/1/0] 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
```

 Copyright © Huawei Technologies Co., Ltd.

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 GigabitEthernet0/1/0
      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 GigabitEthernet0/1/0
      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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
 rip authentication-mode hmac-sha256 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%# 1 200
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
 rip authentication-mode hmac-sha256 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%# 1 200
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
 rip authentication-mode hmac-sha256 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%# 1 200
#
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 GigabitEthernet0/2/0
```

```
 undo shutdown
 ip address 172.16.1.2 255.255.255.0
 rip authentication-mode hmac-sha256 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%# 1 200
#
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 GigabitEthernet0/3/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
 rip authentication-mode hmac-sha256 cipher %^%#c;\wJ4Qi8I1FMGM}KmIK9rha/.D.!$"~0(Ep66z~%^
%# 1 200
#
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
```

1.1.7.14.2 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 1-74**, 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 1-74 Network diagram of configuring routing loop prevention

NOTE

interface1, interface2, interface3, and interface4 in this example represent GE0/1/0, GE0/1/1, GE0/1/2, and GE0/1/3, 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 gigabitethernet0/1/0
[~DeviceA-GigabitEthernet0/1/0] rip split-horizon
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet0/1/1
[*DeviceA-GigabitEthernet0/1/1] rip split-horizon
[*DeviceA-GigabitEthernet0/1/1] quit
[*DeviceA] interface gigabitethernet0/1/2
[*DeviceA-GigabitEthernet0/1/2] rip split-horizon
[*DeviceA-GigabitEthernet0/1/2] quit
[*DeviceA] interface gigabitethernet0/1/3
[*DeviceA-GigabitEthernet0/1/3] rip split-horizon
[*DeviceA-GigabitEthernet0/1/3] 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
 GigabitEthernet0/1/0(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
 GigabitEthernet0/1/1(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
 GigabitEthernet0/1/2(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
GigabitEthernet0/1/3(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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.1 255.255.255.0
 rip version 2 multicast
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.1.1 255.255.0.0
 rip version 2 multicast
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 10.2.1.1 255.255.0.0
 rip version 2 multicast
#
interface GigabitEthernet0/1/3
 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
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
```

```
 rip version 2 multicast
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 192.168.2.1 255.255.255.0
 rip version 2 multicast
#
rip 1
 network 192.168.0.0
return
```

- DeviceC configuration file

```
#
 sysname DeviceC
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
 rip version 2 multicast
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 172.16.1.1 255.255.0.0
 rip version 2 multicast
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 172.17.1.1 255.255.0.0
 rip version 2 multicast
#
interface GigabitEthernet0/1/3
 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
```

1.1.7.14.3 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 1-75**, 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 1-75 Network diagram of configuring RIP to import external routes

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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     GigabitEthernet0/2/0
    192.168.0.1/32  Direct 0    0         D      127.0.0.1       GigabitEthernet0/2/0
    192.168.1.0/24  Direct 0    0         D      192.168.1.1     GigabitEthernet0/1/0
    192.168.1.1/32  Direct 0    0         D      127.0.0.1       GigabitEthernet0/1/0
    192.168.1.2/32  Direct 0    0         D      192.168.1.2     GigabitEthernet0/1/0
```

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     GigabitEthernet0/2/0
  192.168.0.1/32    Direct 0    0        D       127.0.0.1       GigabitEthernet0/2/0
  192.168.1.0/24    Direct 0    0        D       192.168.1.1     GigabitEthernet0/1/0
  192.168.1.1/32    Direct 0    0        D       127.0.0.1       GigabitEthernet0/1/0
  192.168.1.2/32    Direct 0    0        D       192.168.1.2     GigabitEthernet0/1/0
    192.168.2.0/24  RIP    100  4        D       192.168.1.2     GigabitEthernet0/1/0
    192.168.3.0/24  RIP    100  4        D       192.168.1.2     GigabitEthernet0/1/0
    192.168.4.0/24  RIP    100  4        D       192.168.1.2     GigabitEthernet0/1/0
```

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     GigabitEthernet0/2/0
  192.168.0.1/32    Direct 0    0        D       127.0.0.1       GigabitEthernet0/2/0
  192.168.1.0/24    Direct 0    0        D       192.168.1.1     GigabitEthernet0/1/0
  192.168.1.1/32    Direct 0    0        D       127.0.0.1       GigabitEthernet0/1/0
  192.168.1.2/32    Direct 0    0        D       192.168.1.2     GigabitEthernet0/1/0
  192.168.2.0/24    RIP    100  4        D       192.168.1.2     GigabitEthernet0/1/0
  192.168.3.0/24    RIP    100  4        D       192.168.1.2     GigabitEthernet0/1/0
```

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 GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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
#
```

 Copyright © Huawei Technologies Co., Ltd.

```
return
```

1.1.7.14.4 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 1-76**, 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 1-76 Configuring BFD for RIP

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 0/1/0 (2.2.2.1/24), and IP address of GE 0/2/0 (3.3.3.1/24)
- DeviceB: RIP process ID (1), RIP version number (2), IP address of GE 0/1/0 (2.2.2.2/24), IP address of GE 0/2/0 (4.4.4.1/24), and IP address of GE 0/3/0 (172.16.1.1/24)
- DeviceC: RIP process ID (1), RIP version number (2), IP address of GE 0/1/0 (4.4.4.2/24), and IP address of GE 0/2/0 (3.3.3.2/24)
- DeviceD: RIP process ID (1), RIP version number (2), and IP address of GE 0/1/0 (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 1-76** 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         GigabitEthernet0/2/0        RIP    0:0:5
 Number of RIP routes  :2
 2.2.2.2         GigabitEthernet0/1/0        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         GigabitEthernet0/2/0
       3.3.3.0/24  Direct 0    0             D  3.3.3.1         GigabitEthernet0/2/0
       3.3.3.1/32  Direct 0    0             D  127.0.0.1       GigabitEthernet0/2/0
     3.3.3.255/32  Direct 0    0             D  127.0.0.1       GigabitEthernet0/2/0
       2.0.0.0/8   RIP    100  3             D  2.2.2.2         GigabitEthernet0/1/0
       2.2.2.0/24  Direct 0    0             D  2.2.2.1         GigabitEthernet0/1/0
       2.2.2.1/32  Direct 0    0             D  127.0.0.1       GigabitEthernet0/1/0
     2.2.2.255/32  Direct 0    0             D  127.0.0.1       GigabitEthernet0/1/0
     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         GigabitEthernet0/1/0
    172.16.1.0/24  RIP    100  1             D  2.2.2.2         GigabitEthernet0/1/0
       4.4.4.0/24  RIP    100  1             D  3.3.3.2         GigabitEthernet0/2/0
                   RIP    100  1             D  2.2.2.2         GigabitEthernet0/1/0
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 0/1/0. 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  :GigabitEthernet0/1/0
 LocalIp     :2.2.2.1        RemoteIp   :2.2.2.2        BFDState :Up

 Interface :GigabitEthernet0/2/0
 LocalIp     :3.3.3.1        RemoteIp  :3.3.3.2         BFDState :Down
```

Step 4 Verifying the Configuration

Run the **shutdown** command on GigabitEthernet 0/1/0 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 gigabitethernet0/1/0
[~DeviceB-GigabitEthernet0/1/0] shutdown
[*DeviceB-GigabitEthernet0/1/0] 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 :GigabitEthernet0/2/0
 LocalIp     :3.3.3.1        RemoteIp  :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         GigabitEthernet0/2/0
        3.3.3.1/32  Direct 0    0           D  127.0.0.1       GigabitEthernet0/2/0
      3.3.3.255/32  Direct 0    0           D  127.0.0.1       GigabitEthernet0/2/0
      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         GigabitEthernet0/2/0
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 0/2/0.

----End

Configuration Files

- DeviceA configuration file

```
#
 sysname DeviceA
#
bfd
#
interface gigabitethernet0/1/0
 undo shutdown
 ip address 2.2.2.1 255.255.255.0
#
interface gigabitethernet0/2/0
 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 gigabitethernet0/1/0
 undo shutdown
 ip address 2.2.2.2 255.255.255.0
#
interface gigabitethernet0/2/0
 undo shutdown
 ip address 4.4.4.1 255.255.255.0
#
interface gigabitethernet0/3/0
 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 gigabitethernet0/1/0
 undo shutdown
 ip address 4.4.4.2 255.255.255.0
#
interface gigabitethernet0/2/0
 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
#
```

 Copyright © Huawei Technologies Co., Ltd.

```
return
```

- DeviceD configuration file

```
#
 sysname DeviceD
#
interface gigabitethernet0/1/0
 undo shutdown
 ip address 172.16.1.2 255.255.255.0
#
rip 1
 version 2
 network 172.16.0.0
#
return
```

1.1.7.14.5 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 1-77**:

- 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 1-77 Configuring static BFD for RIP

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 0/1/0 (2.2.2.1/24), and IP address of GE 0/2/0 (3.3.3.1/24)
- DeviceB: RIP process ID (1), RIP version number (2), and IP addresses of GE 0/1/0, GE 0/2/0, and GE0/3/0 (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 0/1/0 (4.4.4.2/24), and IP address of GE 0/2/0 (3.3.3.2/24)
- DeviceD: RIP process ID (1), RIP version number (2), and IP address of GE 0/1/0 (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 1-77** 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         GigabitEthernet0/2/0         RIP    0:0:5
 Number of RIP routes  :2
 2.2.2.2         GigabitEthernet0/1/0         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         GigabitEthernet0/2/0
       3.3.3.0/24  Direct 0    0           D   3.3.3.1         GigabitEthernet0/2/0
       3.3.3.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/2/0
     3.3.3.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/2/0
       2.0.0.0/8   RIP    100  3           D   2.2.2.2         GigabitEthernet0/1/0
       2.2.2.0/24  Direct 0    0           D   2.2.2.1         GigabitEthernet0/1/0
       2.2.2.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
     2.2.2.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
     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         GigabitEthernet0/1/0
    172.16.1.0/24  RIP    100  1           D   2.2.2.2         GigabitEthernet0/1/0
       4.4.4.0/24  RIP    100  1           D   3.3.3.2         GigabitEthernet0/2/0
                   RIP    100  1           D   2.2.2.2         GigabitEthernet0/1/0
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 0/1/0. 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 gigabitethernet0/1/0 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 gigabitethernet0/1/0
[~DeviceA-GigabitEthernet0/1/0] rip bfd static
```

Configure static BFD on DeviceB.

```
[~DeviceB] bfd
[*DeviceB-bfd] quit
[*DeviceB] bfd 1 bind peer-ip 2.2.2.1 interface gigabitethernet0/1/0 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 gigabitethernet0/1/0
[~DeviceB-GigabitEthernet0/1/0] 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     GigabitEthernet0/1/0
--------------------------------------------------------------------------------
   Total UP/DOWN Session Number : 1/0
```

Step 4 Verifying the Configuration

Run the **shutdown** command on GigabitEthernet 0/1/0 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 gigabitethernet0/1/0
[~DeviceB-GigabitEthernet0/1/0] shutdown
[*DeviceB-GigabitEthernet0/1/0] 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         GigabitEthernet0/2/0
        3.3.3.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/2/0
      3.3.3.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/2/0
      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         GigabitEthernet0/2/0
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 0/2/0.

----End

Configuration Files

- DeviceA configuration file

```
#
 sysname DeviceA
#
bfd
#
interface gigabitethernet0/1/0
 undo shutdown
 ip address 2.2.2.1 255.255.255.0
 rip bfd static
#
interface gigabitethernet0/2/0
 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 gigabitethernet0/1/0 source-ip 2.2.2.1
 discriminator local 1
 discriminator remote 2
#
return
```

- DeviceB configuration file

```
#
 sysname DeviceB
#
bfd
#
interface gigabitethernet0/1/0
 undo shutdown
 ip address 2.2.2.2 255.255.255.0
 rip bfd static
#
interface gigabitethernet0/2/0
 undo shutdown
 ip address 4.4.4.1 255.255.255.0
#
interface gigabitethernet0/3/0
 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 gigabitethernet0/1/0 source-ip 2.2.2.2
 discriminator local 2
 discriminator remote 1
#
return
```

- DeviceC configuration file

```
#
 sysname DeviceC
#
interface gigabitethernet0/1/0
 undo shutdown
 ip address 4.4.4.2 255.255.255.0
#
interface gigabitethernet0/2/0
 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 gigabitethernet0/1/0
 undo shutdown
 ip address 172.16.1.2 255.255.255.0
#
rip 1
 version 2
 network 172.16.0.0
#
return
```

1.1.8 RIPng Configuration

RIPng is an extension of RIP to support IPv6.

1.1.8.1 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.

1.1.8.2 Configuration Precautions for RIPng

Feature Requirements

None

1.1.8.3 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.

1.1.8.3.1 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.

NOTE

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

1.1.8.3.2 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.

NOTE

- 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

1.1.8.3.3 (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

1.1.8.3.4 (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

1.1.8.3.5 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

1.1.8.4 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.
- **1.1.8.3 Configuring Basic RIPng Functions**

Configuration Procedure

Perform one or more of the following configuration tasks (excluding "Checking the Configuration") as required.

1.1.8.4.1 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

1.1.8.4.2 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

1.1.8.4.3 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

1.1.8.4.4 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

1.1.8.5 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.
- **1.1.8.3 Configuring Basic RIPng Functions**

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.8.5.1 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

1.1.8.5.2 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

1.1.8.5.3 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 **1.1.8.8 Configuring RIPng Timers**.

1.1.8.5.4 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

1.1.8.6 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.
- **1.1.8.3 Configuring Basic RIPng Functions**

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.8.6.1 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.

1.1.8.6.2 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

1.1.8.6.3 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

1.1.8.7 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.
- **1.1.8.3 Configuring Basic RIPng Functions**

Configuration Procedure

Perform one or more of the following configurations as required.

1.1.8.7.1 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

1.1.8.7.2 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

1.1.8.7.3 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**]] *

     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

1.1.8.7.4 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.

1.1.8.7.5 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.

NOTE

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

1.1.8.7.6 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

1.1.8.8 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 **1.1.8.5.3 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.
- **1.1.8.3 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.

1.1.8.9 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**

1.1.8.9.1 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.

 Copyright © Huawei Technologies Co., Ltd.

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

1.1.8.9.2 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.

NOTE

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

1.1.8.9.3 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

1.1.8.10 Maintaining RIPng

RIPng maintenance is implemented through debugging. Debugging, however, affects the system performance.

1.1.8.10.1 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

1.1.8.10.2 Debugging RIPng

A device generates debugging information after you enable debugging of modules in the user view. Debugging information shows the contents of the packets sent or received by the debugged modules.

 Copyright © Huawei Technologies Co., Ltd.

Context

NOTICE

Debugging affects system performance. Therefore, after debugging, run the **undo debugging all** command to disable it immediately.

When a RIPng fault occurs, run any of the following **debugging** commands in the user view to debug RIPng and locate the fault.

Procedure

Step 1 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 debug the specified RIPng process.

Step 2 Run the **debugging ripng miscellaneous** command to debug component-level RIPng information.

----End

1.1.8.11 RIPng Configuration Examples

This section provides RIPng configuration examples.

1.1.8.11.1 Example for Configuring Basic RIPng Functions

This section describes how to configure basic RIPng functions.

Networking Requirements

On the network shown in **Figure 1-78**, 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 1-78 Network diagram of basic RIPng function configuration

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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 gigabitethernet0/1/0
[*DeviceA-GigabitEthernet0/1/0] ipv6 enable
[*DeviceA-GigabitEthernet0/1/0] ripng 1 enable
[*DeviceA-ripng-1] quit
[*DeviceA] commit
```

Configure DeviceB.

```
[*DeviceB] ripng 1
[*DeviceB-ripng-1] quit
[*DeviceB] interface gigabitethernet0/1/0
[*DeviceB-GigabitEthernet0/1/0] ipv6 enable
[*DeviceB-GigabitEthernet0/1/0] ripng 1 enable
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet0/2/0
[*DeviceB-GigabitEthernet0/2/0] ipv6 enable
[*DeviceB-GigabitEthernet0/2/0] ripng 1 enable
[*DeviceB-GigabitEthernet0/2/0] quit
[*DeviceB] interface gigabitethernet0/3/0
[*DeviceB-GigabitEthernet0/3/0] ipv6 enable
[*DeviceB-GigabitEthernet0/3/0] ripng 1 enable
[*DeviceB-GigabitEthernet0/3/0] quit
[*DeviceB] commit
```

Configure DeviceC.

```
[*DeviceC] ripng 1
[*DeviceC-ripng-1] quit
[*DeviceC] interface gigabitethernet0/2/0
[*DeviceC-GigabitEthernet0/2/0] ipv6 enable
[*DeviceC-GigabitEthernet0/2/0] ripng 1 enable
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] commit
```

Configure DeviceD.

```
[*DeviceD] ripng 1
[*DeviceD-ripng-1] quit
[*DeviceD] interface gigabitethernet0/3/0
[*DeviceD-GigabitEthernet0/3/0] ipv6 enable
[*DeviceD-GigabitEthernet0/3/0] ripng 1 enable
[*DeviceD-GigabitEthernet0/3/0] 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 GigabitEthernet0/1/0
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 GigabitEthernet0/1/0
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 GigabitEthernet0/2/0
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 GigabitEthernet0/3/0
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 GigabitEthernet0/1/0
 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 %#%#<}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
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/96
 ripng 1 enable
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/96
 ripng 1 enable
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/3/0
 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
```

1.1.8.11.2 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 1-79**, 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 1-79 Network diagram of configuring routing loop prevention

NOTE

Interfaces 1 and 2 in this example represent GE 0/1/0 and GE 0/2/0, 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 gigabitethernet0/1/0
[~DeviceA-GigabitEthernet0/1/0] ripng split-horizon
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet0/1/1
[*DeviceA-GigabitEthernet0/1/1] ripng split-horizon
[*DeviceA-GigabitEthernet0/1/1] quit
[*DeviceA] interface gigabitethernet0/1/2
[*DeviceA-GigabitEthernet0/1/2] ripng split-horizon
[*DeviceA-GigabitEthernet0/1/2] quit
[*DeviceA] interface gigabitethernet0/1/3
[*DeviceA-GigabitEthernet0/1/3] ripng split-horizon
[*DeviceA-GigabitEthernet0/1/3] 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
 GigabitEthernet0/1/0
    FE80::A0A:200:1
    State : UP, Protocol : RIPNG, MTU : 1440
    Metricin : 0 , Metricout : 1
    Default Route : Disabled
    Poison Reverse : Disabled
    Split Horizon : Enabled
 GigabitEthernet0/1/1(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
 GigabitEthernet0/1/2
    FE80::A0A:200:1
    State : UP, Protocol : RIPNG, MTU : 1440
    Metricin : 0 , Metricout : 1
    Default Route : Disabled
    Poison Reverse : Disabled
    Split Horizon : Enabled
 GigabitEthernet0/1/3
    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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
 ripng 1 enable
#
interface GigabitEthernet0/1/1
 undo shutdown
```

```
 ipv6 enable
 ipv6 address 2001:db8:3::1/96
 ripng 1 enable
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 10.2.1.1 255.255.0.0
#
interface GigabitEthernet0/1/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:5::1/96
 ripng 1 enable
#
ripng 1
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 ripng 1 enable
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
 ripng 1 enable
#
return
```

- DeviceC configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/64
 ripng 1 enable
#
interface GigabitEthernet0/1/1
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:7::1/96
 ripng 1 enable
#
interface GigabitEthernet0/1/2
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:8::1/96
 ripng 1 enable
#
interface GigabitEthernet0/1/3
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:9::1/96
 ripng 1 enable
#
ripng 1
#
return
```

 Copyright © Huawei Technologies Co., Ltd.

1.1.9 IS-IS Configuration

This chapter describes the principles and provides configuration procedures and configuration examples of Intermediate System to Intermediate System (IS-IS).

1.1.9.1 Overview of IS-IS

ISIS protocol can implement the interworking of 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 its Connectionless Network Protocol (CLNP).

To support IP routing, the Internet Engineering Task Force (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. This 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

If IS-IS IPv4 and IS-IS IPv6 implement a feature in the same way, details are not provided in this chapter.

Purpose

As an Interior Gateway Protocol (IGP), IS-IS is used in Autonomous Systems (ASs). IS-IS is a link state protocol, and it uses the Shortest Path First (SPF) algorithm to calculate routes.

1.1.9.2 Configuration Precautions for IS-IS

Feature Requirements

Table 1-14 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. Association between BFD and IS-IS neighbor Up does not support broadcast interfaces or advanced MTs. The association between BFD and IS-IS neighbor Up does not support broadcast interfaces or the advanced MT.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In Encap mode, when the last SID of a Policy is a BSID, traffic fails to be forwarded.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
The NSO has its own configuration database and saves delivered configurations. When a configuration with a must condition, it is checked based on the configuration database saved by the NSO. The configuration can be delivered only when it meets the must condition. 1. When a configuration with the must condition is configured, the matching condition is delivered by the NSO command and can be delivered without repeated delivery or synchronization. 2. When a configuration with the must condition is configured, the matching condition is delivered by the CLI or other tools. The new configuration can be delivered only after synchronization. If synchronization is not performed, the must check may fail. Plan the configuration properly.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
If an IS-IS Level-1-2 router imports a route with a priority lower than IS-IS, ensure that the route is not learned by another level of the current router after route leaking. Otherwise, route flapping occurs. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
When multiple import points exist and different devices are configured with the same system ID, a false loop alarm may be reported.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
When load balancing is used: 1. TI-LFA backup next hop calculation is not supported in load balancing scenarios. 2. FRR is not supported in load balancing scenarios where multiple nodes advertise the same route. 3. FRR calculation is not supported in SR-MPLS BE load balancing scenarios. Therefore, plan the network properly.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
1. The mapping SID does not support priority comparison. 2. The mapping SID cannot be used in anycast FRR. 3. The mapping SID cannot be used in the SR anti-micro-loop function. 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. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In SR and LDP interworking scenarios, FRR does not take effect in the following situations: In the SR-to-LDP direction, one of the Q node and NEs on the path before the Q node does not support SR. In the LDP-to-SR direction, one of the PQ node and NEs on the path before the PQ node does not support LDP. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
SBFD is not supported in SR and LDP interworking scenarios because LDP does not support SBFD. Properly plan the network.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
Association between BFD and IS-IS neighbor status does not support broadcast interfaces or advanced MT interfaces.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.9.3 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.

1.1.9.3.1 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.

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. 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.

  The value of *symbolic-name* is contained in LSP packets and advertised to other IS-IS devices.

  The configured hostname (*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** *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

1.1.9.3.2 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

1.1.9.3.3 (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 receives 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 receives 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 receives 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 by the interface.

         If the cost of the route is greater than 1023, the device can learn only the routes to 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 by 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 a received route is 0xFFFFFFFF.

d. Run **commit**

The configuration is committed.

- Configure the cost for 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** [**topology** *topology-name*] **cost** *cost* [**level-1** | **level-2**]

A cost is configured for 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**

The system view is displayed.

b. Run **isis** [*process-id*]

The IS-IS view is displayed.

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**

The system view is displayed.

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 1-15** lists the interface costs corresponding to different interface bandwidth ranges.

  Table 1-15 Mapping between IS-IS interface costs and interface bandwidth

Cost	Bandwidth Range
60	Interface bandwidth ≤ 10 Mbit/s
50	10 Mbit/s < interface bandwidth ≤ 100 Mbit/s
40	100 Mbit/s < interface bandwidth ≤ 155 Mbit/s
30	155 Mbit/s < interface bandwidth ≤ 622 Mbit/s
20	622 Mbit/s < Interface bandwidth ≤ 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.

----End

1.1.9.3.4 (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**] }

The negotiation mode is specified on the interface.

The **isis ppp-negotiation** command can only be used for the establishment of the neighbor relationships on 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**

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.

NOTE

c and **d** are mutually exclusive. Run the command as needed.

c. Run **isis small-hello**

The interface is configured to send Hello packets without the padding field.

d. Run **isis padding-hello**

The interface is configured to send standard Hello packets with the padding field.

e. Run **commit**

The configuration is committed.

- Configure IS-IS not to check the IP addresses of received Hello packets.

a. Run **system-view**

The system view is displayed.

b. Run **interface** *interface-type interface-number*

The interface view is displayed.

c. Run **isis peer-ip-ignore**

IS-IS is configured not to check the IP addresses of received Hello packets.

d. Run **commit**

The configuration is committed.

----End

1.1.9.3.5 (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

1.1.9.3.6 (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

1.1.9.3.7 (Optional) Enabling the Type-132 TLV in LSPs to Carry the IPv4 Address of Only One IS-IS Interface

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

1.1.9.3.8 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

1.1.9.4 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**.

1.1.9.4.1 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 the **import-route isis level-2 into level-1** [**filter-policy** { *acl-number* | **acl-name** *acl-name* } | **tag** *tag* | **no-sid**] * command.
       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 the **import-route isis level-2 into level-1** [**filter-policy ip-prefix** *ip-prefix-name* | **tag** *tag* | **no-sid**] * command.

- Based on a route-policy:

  Run the **import-route isis level-2 into level-1** [**filter-policy route-policy** *route-policy-name* | **tag** *tag* | **no-sid**] * 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.
  c. 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:
       1) Run the **import-route isis level-1 into level-2** [**filter-policy** { *acl-number* | **acl-name** *acl-name* } | **tag** *tag* | **no-sid**] * command.
       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*] *

          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 the **import-route isis level-1 into level-2** [**filter-policy ip-prefix** *ip-prefix-name* | **tag** *tag* | **no-sid**] * command.

- Based on a route-policy:

  Run the **import-route isis level-1 into level-2** [**filter-policy route-policy** *route-policy-name* | **tag** *tag* | **no-sid**] * 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

1.1.9.4.2 Configuring Rules for Selecting Equal-Cost IPv4 IS-IS Routes

If multiple equal-cost IS-IS routes are available on a network, configure the equal-cost IS-IS routes to load-balance traffic to increase the bandwidth usage of each link, or configure priorities for the equal-cost IS-IS routes to facilitate traffic management.

Context

If there are multiple IS-IS routes with the same cost, choose either of the following methods to use these equal-cost IS-IS routes:

- Configure load balancing for equal-cost IS-IS routes.

  Load balancing increases the link bandwidth usage and prevents network congestion caused by link overload. However, Load balancing may complicate traffic management because traffic will be randomly forwarded.

- Configure priorities for equal-cost IS-IS routes so that only the route with the highest priority is preferentially selected, and the others function as backups.

  The original configurations do not need to be changed. This method facilitates traffic management and improves the network reliability.

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 preference: Routes with lower preference value (higher preference) 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 priorities, routes with smaller System ID are selected for load balancing.
     3. Interface index: If routes have the same priorities and System ID, routes with lower interface index values 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 priorities 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 priority values 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 priority is configured for an equal-cost IS-IS route.

     NOTE

     A smaller value of the *value* parameter indicates a higher priority.
  d. Run **commit**

     The configuration is committed.

----End

1.1.9.4.3 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

1.1.9.4.4 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

1.1.9.4.5 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.

  NOTE

  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

1.1.9.4.6 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

1.1.9.4.7 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

1.1.9.4.8 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

1.1.9.5 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.

1.1.9.5.1 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.

To prefer a route discovered by IS-IS, configure a higher priority value for IS-IS. In addition, you can configure a routing policy to increase the priority of specified IS-IS routes, without affecting route selection.

Procedure

- Configure the IS-IS priority value.
  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.

NOTE

A smaller *preference* value indicates a higher preference.

d. Run **commit**

The configuration is committed.

- Configure a priority 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 { { **route-policy** *route-policy-name* | **route-filter** *route-filter-name* } | *preference* } *

The priority value is configured for the specified IS-IS routes.

NOTE

preference takes effect only on IS-IS routes that match the specified routing policy.

d. Run **commit**

The configuration is committed.

----End

1.1.9.5.2 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 or advertise external routes that meet specified conditions to the IS-IS routing domain.

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 the **filter-policy** { *acl-number* | **acl-name** *acl-name* } **export** [*protocol* [*process-id*]] command.
       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

1.1.9.5.3 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

1.1.9.6 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**.

1.1.9.6.1 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.

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

1.1.9.6.2 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 NE40E 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.

     NOTE

     Configure **Level-1** and **Level-2** only when a broadcast interface is specified.
  d. Run **commit**

     The configuration is committed.
- 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 synchronize all LSPs in the areas, IS-IS regularly transmits all the current LSPs to neighbors.

Ensure that the LSP update interval is at least 300s shorter than the maximum LSP lifetime so that new LSPs can reach all devices in an area before existing LSPs expire.

NOTE

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

1.1.9.6.3 Setting 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

 Copyright © Huawei Technologies Co., Ltd.

1.1.9.6.4 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 NE40E 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

1.1.9.6.5 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

1.1.9.6.6 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

1.1.9.7 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.

 Copyright © Huawei Technologies Co., Ltd.

- 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.

1.1.9.8 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.

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 Flex-Algo algorithm, the difference in delay between links is generally different and may be very small. However, according to the existing SPF algorithm, only one optimal path can be generated even if the difference is very small, and load balancing cannot be implemented within a certain delay tolerance range. Consequently, link resources on a network cannot be fully utilized. To solve this problem to the greatest extent, normalization can be performed on the link delays with a very small difference or with 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.

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 delay information is enabled.

Step 7 (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 8 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.

1.1.9.9 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.

1.1.9.10 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** *discr-value* command.
- To set the remote discriminator, run the **discriminator remote** *discr-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.

1.1.9.11 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:

1.1.9.11.1 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

1.1.9.11.2 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.

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 **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) Run **bfd all-interfaces** { **min-rx-interval** *receive-interval* | **min-tx-interval** *transmit-interval* | **detect-multiplier** *multiplier-value* | **frr-binding** | **tos-exp** *tosexp-value* } *

BFD parameters are configured to establish BFD sessions.

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** <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 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

1.1.9.11.3 (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

1.1.9.11.4 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

1.1.9.12 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

1.1.9.13 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:

1.1.9.13.1 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

1.1.9.13.2 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.

NOTE

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

1.1.9.13.3 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

1.1.9.14 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.

NOTE

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.

1.1.9.15 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.

1.1.9.16 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:

1. Create IPv6 IS-IS processes.
2. 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.

1.1.9.16.1 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.

     If dynamic host name mapping is configured on an IS-IS network, the mappings on the network override those 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.

----End

1.1.9.16.2 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.
  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 **ipv6 enable**

     IPv6 is enabled on the interface.
  e. 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.
  f. 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.

b. Run **commit**

The configuration is committed.

----End

1.1.9.16.3 (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 the cost 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 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 1-16 lists the interface costs corresponding to different interface bandwidth ranges.

       Table 1-16 Mapping between IS-IS interface costs and interface bandwidth

Cost	Bandwidth Range
60	Interface bandwidth ≤ 10 Mbit/s
50	10 Mbit/s < interface bandwidth ≤ 100 Mbit/s
40	100 Mbit/s < interface bandwidth ≤ 155 Mbit/s
30	155 Mbit/s < interface bandwidth ≤ 622 Mbit/s
20	622 Mbit/s < Interface bandwidth ≤ 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

1.1.9.16.4 (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**

OSICP negotiation status check is configured on the 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 IS-IS not to check the IP addresses of received Hello packets.

a. Run **system-view**

The system view is displayed.

b. Run **interface** *interface-type interface-number*

The interface view is displayed.

c. Run **isis peer-ip-ignore**

IS-IS is configured not to check the IP addresses of received Hello packets.

d. Run **commit**

The configuration is committed.

----End

1.1.9.16.5 (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

1.1.9.16.6 (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**]] *

Enable LSP fragment extension on an IS-IS device in a specified mode and at a specified level.

----End

1.1.9.16.7 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 features have been configured.

Procedure

Step 1 Run the **display isis name-table** [*process-id* | **vpn-instance** *vpn-instance-name*] command to check the mapping from the name of the local device to 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

1.1.9.17 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**

  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 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**.

1.1.9.17.1 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.
  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:

     - Configure a basic ACL:
       1) 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.
       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.

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:

- 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.

- 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.

c. 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

1.1.9.17.2 Configuring Rules for Selecting Equal-Cost IPv6 IS-IS Routes

If multiple equal-cost IS-IS routes are available on a network, configure the equal-cost IS-IS routes to load-balance traffic to increase the bandwidth usage of each link, or configure priorities for the equal-cost IS-IS routes to facilitate traffic management.

Context

If there are multiple IPv6 IS-IS routes with the same cost, configure load balancing for equal-cost IPv6 IS-IS routes.

Load balancing increases the link bandwidth usage and prevents network congestion caused by link overload. However, Load balancing may complicate traffic management because traffic will be randomly forwarded.

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 maximum load-balancing** *number*

The maximum number of equal-cost IPv6 IS-IS routes for load balancing is set.

NOTE

If the actual equal-cost routes outnumber the value specified in the **ipv6 maximum load-balancing** command, routes are selected for load balancing based on the following rules:

1. Route preference: Routes with lower preference value (higher preference) are selected for load balancing.
2. Next hop System ID: If routes have the same priorities, routes with smaller System ID are selected for load balancing.
3. Interface index: If routes have the same priorities and System ID, routes with lower interface index values are selected for load balancing.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.9.17.3 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

1.1.9.17.4 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

1.1.9.17.5 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.

  NOTE

  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

1.1.9.17.6 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

1.1.9.17.7 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 in the IS-IS LSDB.

----End

1.1.9.18 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.

1.1.9.18.1 Configuring a Priority Value for IPv6 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 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.

To prefer a route discovered by IS-IS, configure a higher priority value for IS-IS. In addition, you can configure a routing policy to increase the priority of specified IS-IS routes, without affecting route selection.

Procedure

- Configure the IS-IS priority value.
  a. Run **system-view**

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

The IS-IS view is displayed.

c. Run **ipv6 preference** *preference*

The IS-IS priority value is configured.

NOTE

A smaller *preference* value indicates a higher preference.

d. Run **commit**

The configuration is committed.

- Configure a priority 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 **ipv6 preference** { { **route-policy** *route-policy-name* | **route-filter** *route-filter-name* } | *preference* } *

The priority value is configured for the specified IS-IS routes.

NOTE

preference takes effect only on IS-IS routes that match the specified routing policy.

d. Run **commit**

The configuration is committed.

----End

1.1.9.18.2 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 or advertise external routes that meet specified conditions to the IS-IS routing domain.

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

1.1.9.18.3 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

1.1.9.19 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**.

1.1.9.19.1 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.

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

1.1.9.19.2 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 NE40E 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.

     NOTE

     Configure **Level-1** and **Level-2** only when a broadcast interface is specified.
  d. Run **commit**

     The configuration is committed.
- 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 synchronize all LSPs in the areas, IS-IS regularly transmits all the current LSPs to neighbors.

     Ensure that the LSP update interval is at least 300s shorter than the maximum LSP lifetime so that new LSPs can reach all devices in an area before existing LSPs expire.

     NOTE

     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

1.1.9.19.3 Setting the SPF Calculation Interval

To improve the fault location efficiency on an IS-IS network and prevent SPF calculation from consuming excessive system resources, set the SPF calculation interval to a proper value.

Context

A network change always triggers IS-IS to perform SPF calculation. 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

1.1.9.19.4 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 NE40E 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

1.1.9.19.5 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 suppress-reachability**

An IS-IS interface is configured to suppress the advertisement of interface addresses.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.9.19.6 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

1.1.9.20 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.

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 Flex-Algo algorithm, the difference in delay between links is generally different and may be very small. However, according to the existing SPF algorithm, only one optimal path can be generated even if the difference is very small, and load balancing cannot be implemented within a certain delay tolerance range. Consequently, link resources on a network cannot be fully utilized. To solve this problem to the greatest extent, normalization can be performed on the link delays with a very small difference or with 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*] **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**]

IPv6 delay advertisement is configured.

Step 8 (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 9 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.

1.1.9.21 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.

1.1.9.22 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.

NOTE

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.

1.1.9.23 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.

1.1.9.24 Disabling Routing Loop Detection for Routes Imported to IS-IS

Context

By default, routing loop detection is enabled for routes imported to IS-IS. By default, routing loop detection is enabled for routes imported to IS-IS because routing loops may occur when an IS-IS process imports routes. If such detection is enabled 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.

 Copyright © Huawei Technologies Co., Ltd.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 (Optional) Run **clear route loop-detect isis alarm-state**

The routing loop detection alarm state exits, and related alarms are cleared.

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.

Step 3 Run **route loop-detect isis disable**

Routing loop detection is disabled for imported IS-IS routes.

NOTE

To reduce the impact of routing loops on services, you are advised to keep routing loop detection enabled for routes imported to IS-IS.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.9.25 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.

NOTE

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.

1.1.9.26 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:

1.1.9.26.1 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

1.1.9.26.2 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.

 Copyright © Huawei Technologies Co., Ltd.

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

1.1.9.26.3 (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

1.1.9.26.4 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

1.1.9.27 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

1.1.9.28 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:

1.1.9.28.1 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 Run **ipv6 topology** *topology-name* **topology-id** *topology-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 **ipv6 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 features for IPv6 topology instances using the following commands:

- To set the priority of 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 an IS-IS device to import routes from other routing protocols in the IS-IS view, run the **ipv6 import-route** command.
- 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 8 Run **commit**

The configuration is committed.

----End

1.1.9.28.2 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

1.1.9.28.3 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

1.1.9.29 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.

1.1.9.29.1 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.

 Copyright © Huawei Technologies Co., Ltd.

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

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip topology** *topology-name*

An IPv4 topology is created.

Step 3 Run **ipv6 topology** *topology-name*

An IPv6 topology is created.

Step 4 Run **isis** [*process-id*]

The IS-IS view is displayed.

Step 5 Run **cost-style** { **wide** | **wide-compatible** }

The cost style of packets to be sent and accepted by the Router is set to **wide** or **wide-compatible**.

NOTE

Go to Step 6 or Step 7 as required.

- If the Router supports only IPv4 services, go to Step 6.
- If the Router supports only IPv6 services, go to Step 7.
- If the Router supports both IPv4 and IPv6 services, perform the following steps.

Step 6 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 instance view is displayed.

Step 7 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 instance view is displayed.

Step 8 (Optional) Configure IS-IS parameters for the 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 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.

Step 9 Run **commit**

The configuration is committed.

----End

1.1.9.29.2 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

1.1.9.29.3 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

1.1.9.30 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 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*]

The IS-IS view is displayed.

Step 3 Run **ipv6 summary** *ipv6-address prefix-length* [**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.

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.

1.1.9.31 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.

 Copyright © Huawei Technologies Co., Ltd.

- **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** *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**

The previous view is displayed.

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.

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.

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**]

The solution of selecting a backup path for IPv6 IS-IS Auto FRR is set.

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.

1.1.9.32 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 load balancing. If a link failure occurs and the bandwidth of the other links in the group is not sufficient enough 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 link group view is displayed.

Step 4 Run **cost-offset** { *cost* | **max-reachable** | **maximum** }

A link cost offset is configured.

NOTE

If **maximum** is specified in the command, the maximum value (16777215) is used as the link cost of the member links, and the routes along these links are used to transmit TE information, not selected for traffic forwarding.

Step 5 Run **min-members** *min-num*

A minimum number of member links is specified. If the number of available member links in the link group falls below *min-num*, the configured link cost offset is added to the link cost of these member links.

Step 6 Run **revert-members** *revert-num*

A number is specified. After the number of available member links in the link group reaches or exceeds *revert-num* the original link cost of these member links is restored.

NOTE

The value of *revert-number* must be greater than or equal to that of *min-number*.

Step 7 Run **quit**

Exit from the link group view.

Step 8 Run **quit**

The IS-IS view is displayed.

Step 9 Run **interface** *interface-type interface-number*

The interface view is displayed.

Step 10 Run **isis link-group** *group-name* [**level-1** | **level-2**]

The interface is bound to the link group.

NOTE

To bind multiple interfaces to the link group, repeat Steps 9 and 10.

Step 11 Run **commit**

The configuration is committed.

----End

Checking the Configurations

An IS-IS link group has been configured.

- Run the display isis link-group [process-id][group-name] command to check information about an IS-IS link group.
- Run the **display isis link-group interface** *interface-type interface-number* command to check information about the link group to which the interface has been bound.

1.1.9.33 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 multicast and a Multiprotocol Label Switching (MPLS) TE tunnel are configured on a network, the TE tunnel may send multicast packets which are transparent to the Router through which the tunnel passes. As a result, the Router cannot generate multicast forwarding entries. To resolve this problem, enable IS-IS local MT and Interior Gateway Protocol (IGP) Shortcut. If the outbound interface of a calculated route is an IGP Shortcut-enabled TE tunnel interface, the routing

management (RM) module creates an independent Multicast IGP (MIGP) routing table. Multicast packets are then forwarded along the routes in the MIGP routing table.

NOTE

IS-IS local MT takes effect only in public network IS-IS instances.

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.

1.1.9.33.1 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

1.1.9.33.2 (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

1.1.9.33.3 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

1.1.9.34 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

1.1.9.34.1 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.

 Copyright © Huawei Technologies Co., Ltd.

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.
  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:

     - **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

1.1.9.34.2 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.

NOTE

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

1.1.9.34.3 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

1.1.9.35 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.9.36 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 packets of non-IS-IS neighbors 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

1.1.9.37 Maintaining IS-IS

Maintaining IS-IS involves resetting IS-IS, clearing IS-IS statistics, and debugging IS-IS.

1.1.9.37.1 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

1.1.9.37.2 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

1.1.9.37.3 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

1.1.9.37.4 Disabling IS-IS CPU Overload Control

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.

NOTE

To minimize the impact of CPU overload upon services, you are advised not to disable IS-IS CPU overload control.

Step 3 Run **commit**

The configuration is committed.

----End

1.1.9.37.5 (Optional) Configuring IS-IS to Discard a Specified LSP

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** *lspid* }

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** *lspid* } 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

1.1.9.38 IS-IS Configuration Examples

This section describes IS-IS configuration examples.

1.1.9.38.1 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 1-80**:

- 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 1-80 Configuring basic IS-IS functions

Device Name	Interface	IP Address
Device A	GE0/1/0	10.1.1.2/24
Device B	GE0/1/0	10.1.2.2/24
Device C	GE0/1/0	10.1.1.1/24
	GE0/2/0	10.1.2.1/24
	GE0/3/0	192.168.0.1/24
Device D	GE0/1/0	192.168.0.2/24
	GE0/2/0	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.

This example assumes that you know the configuration method and no details are provided here, for detailed configurations, see **Figure 1-80** and configuration files in this example.

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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable 1
[*DeviceC-GigabitEthernet0/1/0] isis circuit-type p2p
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis enable 1
[*DeviceC-GigabitEthernet0/2/0] isis circuit-type p2p
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] interface gigabitethernet 0/3/0
[*DeviceC-GigabitEthernet0/3/0] isis enable 1
[*DeviceC-GigabitEthernet0/3/0] isis circuit-type p2p
```

```
[*DeviceC-GigabitEthernet0/3/0] commit
[~DeviceC-GigabitEthernet0/3/0] 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 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis enable 1
[*DeviceD-GigabitEthernet0/2/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet0/2/0] quit
[*DeviceD] interface gigabitethernet 0/1/0
[*DeviceD-GigabitEthernet0/1/0] isis enable 1
[*DeviceD-GigabitEthernet0/1/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet0/1/0] commit
[~DeviceD-GigabitEthernet0/1/0] 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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] isis authentication-mode hmac-sha256 key-id 1 cipher YsHsjx_202206
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure Device C.

```
[~DeviceC] interface gigabitethernet 0/1/0
[~DeviceC-GigabitEthernet0/1/0] isis authentication-mode hmac-sha256 key-id 1 cipher YsHsjx_202206
[*DeviceC-GigabitEthernet0/1/0] commit
[~DeviceC-GigabitEthernet0/1/0] 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   0xbf7d       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     0/1/0
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   0xbf7d       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     0/1/0
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   0xbf7d        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     0/1/0
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    GE0/1/0         Direct          D/-/L/-/-
10.1.2.0/24        20        NULL    GE0/1/0         10.1.1.1        A/-/-/-/-
192.168.0.0/24     20        NULL    GE0/1/0         10.1.1.1        A/-/-/-/-
0.0.0.0/0          10        NULL    GE0/1/0         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    GE0/1/0         Direct          D/-/L/-/-
 10.1.2.0/24        10        NULL    GE0/2/0         Direct          D/-/L/-/-
 192.168.0.0/24     10        NULL    GE0/3/0         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    GE0/3/0         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    GE0/3/0         Direct          D/-/L/-/-
 10.1.1.0/24        20        NULL    GE0/3/0         192.168.0.1     A/-/-/-/-
 10.1.2.0/24        20        NULL    GE0/3/0         192.168.0.1     A/-/-/-/-
 172.16.0.0/16      10        NULL    GE0/2/0         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 GigabitEthernet0/1/0
 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;\wJ4Qi8I1FMGM}KmIK9rha/.D.!
$"~0(Ep66z~%^%#
#
return
```

- Device B configuration file

```
#
sysname DeviceB
#
isis 1
 is-level level-1
network-entity 10.0000.0000.0002.00
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
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;\wJ4Qi8I1FMGM}KmIK9rha/.D.!
$"~0(Ep66z~%^%#
```

```
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
 isis enable 1
 isis circuit-type p2p
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
 isis enable 1
 isis circuit-type p2p
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 172.16.1.1 255.255.0.0
 isis enable 1
 isis circuit-type p2p
#
return
```

1.1.9.38.2 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 1-81**:

- 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 Device A functions as a Level-1-2 DIS (DR).

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-81 Configuring IS-IS DIS election

Device Name	Interface	IP Address
Device A	GE0/1/0	10.1.1.1/24
Device B	GE0/1/0	10.1.1.2/24
Device C	GE0/1/0	10.1.1.3/24
Device D	GE0/1/0	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 0/1/0 on Device A.

```
[~DeviceA] display arp interface gigabitethernet 0/1/0
ARP timeout:1200s
IP ADDRESS      MAC ADDRESS     EXPIRE(M) TYPE       INTERFACE   VPN-INSTANCE
                                VLAN/CEVLAN PVC
-------------------------------------------------------------------------------
10.1.1.1        00e0-fc10-afec          I          GE0/1/0
-------------------------------------------------------------------------------
Total:1         Dynamic:0       Static:0    Interface:1    Remote:0
Redirect:0
```

Display the MAC address of GigabitEthernet 0/1/0 on Device B.

```
[~DeviceB] display arp interface gigabitethernet 0/1/0
ARP timeout:1200s
IP ADDRESS      MAC ADDRESS     EXPIRE(M) TYPE       INTERFACE   VPN-INSTANCE
                                VLAN/CEVLAN PVC
-------------------------------------------------------------------------------
10.1.1.2        00e0-fccd-acdf          I          GE0/1/0
-------------------------------------------------------------------------------
Total:1         Dynamic:0       Static:0    Interface:1    Remote:0
Redirect:0
```

Display the MAC address of GigabitEthernet 0/1/0 on Device C.

```
[~DeviceC] display arp interface gigabitethernet 0/1/0
ARP timeout:1200s
IP ADDRESS      MAC ADDRESS     EXPIRE(M) TYPE       INTERFACE   VPN-INSTANCE
                                VLAN/CEVLAN PVC
-------------------------------------------------------------------------------
10.1.1.3        00e0-fc50-25fe          I          GE0/1/0
-------------------------------------------------------------------------------
Total:1         Dynamic:0       Static:0    Interface:1    Remote:0
Redirect:0
```

Display the MAC address of GigabitEthernet 0/1/0 on Device D.

```
[~DeviceD] display arp interface gigabitethernet 0/1/0
ARP timeout:1200s
IP ADDRESS      MAC ADDRESS     EXPIRE(M) TYPE       INTERFACE   VPN-INSTANCE
                                VLAN/CEVLAN PVC
-------------------------------------------------------------------------------
10.1.1.4        00e0-fcf0-23ed          I          GE0/1/0
-------------------------------------------------------------------------------
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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable 1
[*DeviceC-GigabitEthernet0/1/0] commit
[~DeviceC-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] isis enable 1
[*DeviceD-GigabitEthernet0/1/0] commit
[~DeviceD-GigabitEthernet0/1/0] 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  GE0/1/0            0000.0000.0001.01  Up   30s      L1(L1L2) 64
0000.0000.0003  GE0/1/0            0000.0000.0001.01  Up   30s      L1       64
0000.0000.0002  GE0/1/0            0000.0000.0001.01  Up   30s      L2(L1L2) 64
0000.0000.0004  GE0/1/0            0000.0000.0001.01  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
 GE0/1/0            001         Up            Mtu:Up/Lnk:Dn/IP:Dn 1497 L1/L2 No/No
```

Display information about the IS-IS interface on Device B.

```
[~DeviceB] display isis interface
                    Interface information for ISIS(1)
                    ---------------------------------
 Interface          Id      IPV4.State          IPV6.State      MTU  Type  DIS
 GE0/1/0            001         Up            Mtu:Up/Lnk:Dn/IP:Dn 1497 L1/L2 Yes/No
```

Display information about the IS-IS interface on Device D.

```
[~DeviceD] display isis interface
                    Interface information for ISIS(1)
                    ---------------------------------
 Interface          Id      IPV4.State          IPV6.State      MTU  Type  DIS
 GE0/1/0            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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] isis dis-priority 100
[*DeviceA-GigabitEthernet0/1/0] commit
```

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  GE0/1/0           0000.0000.0001.01  Up   27s      L1(L1L2) 64
0000.0000.0003  GE0/1/0           0000.0000.0001.01  Up   27s      L1       64
0000.0000.0002  GE0/1/0           0000.0000.0001.01  Up   27s      L2(L1L2) 64
0000.0000.0004  GE0/1/0           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] display isis interface
                    Interface information for ISIS(1)
                    ---------------------------------
 Interface         Id      IPV4.State          IPV6.State      MTU  Type  DIS
 GE0/1/0           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, Device A becomes a Level-1-2 DIS (DR) immediately, and its pseudo node is 0000.0000.0001.01.

Display information about IS-IS neighbors and the IS-IS interface on Device B.

```
[~DeviceB] display isis peer
                    Peer information for ISIS(1)

  System Id     Interface         Circuit Id       State HoldTime Type     PRI
--------------------------------------------------------------------------------
0000.0000.0001  GE0/1/0           0000.0000.0001.01  Up   26s      L1(L1L2) 100
0000.0000.0003  GE0/1/0           0000.0000.0001.01  Up   26s      L1       64
0000.0000.0001  GE0/1/0           0000.0000.0001.01  Up   26s      L2(L1L2) 100
0000.0000.0004  GE0/1/0           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
 GE0/1/0           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  GE0/1/0           0000.0000.0001.01  Up   20s      L2       100
```

```
0000.0000.0002  GE0/1/0          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
 GE0/1/0          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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ip address 10.1.1.4 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.3 Example for Configuring IS-IS Load Balancing

This section describes how to configure IS-IS load balancing.

Networking Requirements

In **Figure 1-82**:

- 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 1-82 Configuring IS-IS load balancing

Device Name	Interface	IP Address
Device A	GE0/3/0	172.16.1.1/24
	GE0/1/0	10.1.1.1/24
	GE0/2/0	10.1.2.1/24
Device B	GE0/1/0	10.1.1.2/24
	GE0/2/0	192.168.0.1/24
Device C	GE0/1/0	10.1.2.2/24
	GE0/2/0	192.168.1.1/24
Device D	GE0/3/0	172.17.1.1/24
	GE0/1/0	192.168.0.2/24

 Copyright © Huawei Technologies Co., Ltd.

Device Name	Interface	IP Address
	GE0/2/0	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.
5. (Optional) Configure the priority for equal-cost routes 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
- Priority (1 in this example) of equal-cost routes on 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. For configuration details, see **Configuration Files** in this section.

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    GE0/2/0         10.1.2.2        A/-/-/-/-
 10.1.1.0/24         10        NULL    GE0/1/0         Direct          D/-/L/-/-
 172.16.1.0/24       10        NULL    GE0/3/0         Direct          D/-/L/-/-
 172.17.1.0/24       30        NULL    GE0/1/0         10.1.1.2        A/-/-/-/-
 10.1.2.0/24         10        NULL    GE0/2/0         Direct          D/-/L/-/-
 192.168.0.0/24      20        NULL    GE0/1/0         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    GE0/2/0         10.1.2.2        A/-/-/-/-
 10.1.1.0/24         10        NULL    GE0/1/0         Direct          D/-/L/-/-
 172.16.1.0/24       10        NULL    GE0/3/0         Direct          D/-/L/-/-
 172.17.1.0/24       30        NULL    GE0/1/0         10.1.1.2        A/-/-/-/-
                               GE0/2/0         10.1.2.2
 10.1.2.0/24         10        NULL    GE0/2/0         Direct          D/-/L/-/-
 192.168.0.0/24      20        NULL    GE0/1/0         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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ip address 10.1.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.2.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
```

 Copyright © Huawei Technologies Co., Ltd.

```
 ip address 192.168.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 172.17.1.1 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.4 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 1-83**:

- 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 1-83 Configuring IS-IS to interact with BGP

Device Name	Interface	IP Address
Device A	GE0/1/0	10.1.1.1/24
Device B	Loopback0	192.168.0.1/32
	GE0/1/0	10.1.1.2/24
	GE0/2/0	10.2.1.1/24
Device C	Loopback0	192.168.0.2/32
	GE0/1/0	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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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
```

 Copyright © Huawei Technologies Co., Ltd.

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        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.1.1.255/32  Direct 0    0           D   10.1.1.2        GigabitEthernet0/1/0
      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  74           D   10.1.1.2        GigabitEthernet0/1/0
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        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.1.1.255/32  Direct 0    0           D   10.1.1.2        GigabitEthernet0/1/0
       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        GigabitEthernet0/1/0
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        GigabitEthernet0/1/0
      10.2.1.0/24  Direct  0    0           D   10.2.1.2        GigabitEthernet0/1/0
      10.2.1.2/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.2.1.255/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
      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        GigabitEthernet0/1/0
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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 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
#
route-policy RTC permit node 0
 if-match as-path-filter 1
 apply cost 20
#
ip as-path-filter 1 permit 65009
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/1/0
 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
```

1.1.9.38.5 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 1-84**:

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 1-84 Networking for configuring IS-IS multi-instance processes

NOTE

Interfaces 1, 2, and 3 stand for GE 0/1/0, GE 0/2/0, and GE 0/3/0, respectively.

Device Name	Interface	IP Address
Device A	GE 0/1/0	10.1.1.2/24
	GE 0/2/0	192.168.4.4/24
	GE 0/3/0	192.168.5.5/24
Device B	GE 0/1/0	10.1.1.1/24
	GE 0/2/0	192.168.0.1/24
	GE 0/3/0	192.168.1.1/24
Device C	GE 0/1/0	192.168.3.1/24
	GE 0/2/0	192.168.4.2/24
Device D	GE 0/1/0	192.168.3.2/24
	GE 0/2/0	192.168.1.2/24
Device E	GE 0/1/0	192.168.2.4/24

Device Name	Interface	IP Address
	GE 0/2/0	192.168.5.4/24
Device F	GE 0/1/0	192.168.2.5/24
	GE 0/2/0	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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] isis enable 1
[~DeviceA-GigabitEthernet0/1/0] isis enable 2
[~DeviceA-GigabitEthernet0/1/0] isis process-id 1 cost 5
[~DeviceA-GigabitEthernet0/1/0] isis process-id 2 cost 15
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure DeviceB.

```
[~DeviceB] interface gigabitethernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] isis enable 1
[~DeviceB-GigabitEthernet0/1/0] isis enable 2
[~DeviceB-GigabitEthernet0/1/0] isis process-id 1 cost 5
[~DeviceB-GigabitEthernet0/1/0] isis process-id 2 cost 15
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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 0/1/0
                    Interface information for ISIS(1)
                    --------------------------------
 Interface    Id      IPV4.State          IPV6.State        MTU  Type  DIS
 GE0/1/0      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
GE0/1/0      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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.4.4 255.255.255.0
 isis enable 2
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.3.1 255.255.255.0
 isis enable 2
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.3.2 255.255.255.0
 isis enable 2
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.4 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.5 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.0.3 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.6 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 1-85:

- 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 1-85 Configuring static BFD for IS-IS

Device Name	Interface	IP Address
Device A	GE0/1/0	10.1.1.1/24
Device B	GE0/1/0	10.1.1.2/24
	GE0/2/0	10.2.1.1/24
Device C	GE0/1/0	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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] isis enable 1
[*DeviceB-GigabitEthernet0/2/0] commit
[~DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable 1
[*DeviceC-GigabitEthernet0/1/0] commit
[~DeviceC-GigabitEthernet0/1/0] 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 GE0/1/0          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.

```
[~DeviceA] display isis route
                    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    GE0/1/0         Direct         D/-/L/-
 10.2.1.0/24         20        NULL    GE0/1/0         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
```

Step 3 Configure BFD.

Enable BFD and configure a BFD session on Device A.

```
[~DeviceA] bfd
[*DeviceA-bfd] quit
[*DeviceA] bfd atob bind peer-ip 10.1.1.2 interface gigabitethernet0/1/0
[*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 btoa bind peer-ip 10.1.1.1 interface gigabitethernet0/1/0
[*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    GigabitEthernet0/1/0
--------------------------------------------------------------------------------
    Total UP/DOWN Session Number : 1/0
```

Step 4 Enable static BFD on the IS-IS interface.

Configure Device A.

```
[~DeviceA] interface gigabitethernet 0/1/0
[~DeviceA-GigabitEthernet0/1/0] isis bfd static
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
[~DeviceA] quit
```

Configure Device B.

```
[~DeviceB] interface gigabitethernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] isis bfd static
[*DeviceB-GigabitEthernet0/1/0] commit
```

Step 5 Verify the configuration.

Enable debugging on Device A.

```
<DeviceA> debugging isis adjacency
```

```
<DeviceA> debugging isis circuit-information
<DeviceA> terminal debugging
```

Run the **shutdown** command on Gigabit Ethernet 0/1/0 of Device D to simulate a link fault.

```
[~DeviceB-GigabitEthernet0/1/0] shutdown
[*DeviceB-GigabitEthernet0/1/0] 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/-/-/alarmI
D:0x08960007/VS=0:ISIS adjacency state change. (SysInstance=1, SysLevel=1, CircI
ndex=2, CircIfIndex=20, LspId=2222.2222.2222.00.00, AdjState=1, IfIndex=20, IfNa
me=GE0/1/0, 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
 isis enable 1
 isis bfd static
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.2.1.2 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.7 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 1-86**:

- 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 1-86 Configuring dynamic BFD for IS-IS

Device Name	Interface	IP Address
Device A	GE0/1/0	1.1.1.1/24
	GE0/2/0	3.3.3.1/24
Device B	GE0/1/0	2.2.2.2/24
	GE0/2/0	3.3.3.2/24
	GE0/3/0	172.16.1.1/24
Device C	GE0/1/0	1.1.1.2/24
	GE0/2/0	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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/2/0
[*DeviceA-GigabitEthernet0/2/0] isis enable 1
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] isis enable 1
[*DeviceB-GigabitEthernet0/2/0] quit
[*DeviceB] interface gigabitethernet 0/3/0
[*DeviceB-GigabitEthernet0/3/0] isis enable 1
[*DeviceB-GigabitEthernet0/3/0] commit
[~DeviceB-GigabitEthernet0/3/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable 1
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis enable 1
[*DeviceC-GigabitEthernet0/2/0] commit
[~DeviceC-GigabitEthernet0/2/0] 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  GE0/2/0          0000.0000.0002.01 Up   9s      L2      64
0000.0000.0003  GE0/1/0          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         GigabitEthernet0/1/0
        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         GigabitEthernet0/1/0
        3.3.3.0/24  Direct  0    0           D   3.3.3.1         GigabitEthernet0/2/0
        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
```

```
    172.16.1.0/24 ISIS   15   20         D  3.3.3.2       GigabitEthernet0/2/0
```

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 0/2/0
[~DeviceA-GigabitEthernet0/2/0] isis cost 5
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] quit
```

Configure Device B.

```
[~DeviceB] interface gigabitethernet 0/2/0
[~DeviceB-GigabitEthernet0/2/0] isis cost 5
[*DeviceB-GigabitEthernet0/2/0] commit
[~DeviceB-GigabitEthernet0/2/0] 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 : GE0/2/0
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 : GE0/1/0
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 0/2/0 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 0/2/0
[~DeviceA-GigabitEthernet0/2/0] isis bfd enable
[*DeviceA-GigabitEthernet0/2/0] isis bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] quit
```

On GE 0/2/0 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 0/2/0
[~DeviceB-GigabitEthernet0/2/0] isis bfd enable
[*DeviceB-GigabitEthernet0/2/0] isis bfd min-tx-interval 100 min-rx-interval 100 detect-multiplier 4
[*DeviceB-GigabitEthernet0/2/0] commit
[~DeviceB-GigabitEthernet0/2/0] 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 : GE0/2/0
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 : GE0/1/0
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 0/2/0 on Device B to simulate the fault on the primary link.

```
[~DeviceB] interface gigabitethernet 0/2/0
[~DeviceB-GigabitEthernet0/2/0] shutdown
[*DeviceB-GigabitEthernet0/2/0] 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        GigabitEthernet0/1/0
      1.1.1.1/32  Direct 0    0           D  127.0.0.1      GigabitEthernet0/1/0
    1.1.1.255/32  Direct 0    0           D  127.0.0.1      GigabitEthernet0/1/0
      2.2.2.0/24  ISIS   15   20          D  1.1.1.2        GigabitEthernet0/1/0
    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        GigabitEthernet0/1/0
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 : GE0/1/0
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 GigabitEthernet0/1/0
 undo shutdown
 ip address 1.1.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 2.2.2.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
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 GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 1.1.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 2.2.2.1 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.8 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 1-87 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-87 Configuring dynamic BFD for IPv6 IS-IS

Device Name	Interface	IP Address
Device S	GE0/1/0	2001:db8:1::1/64
	GE0/2/0	2001:db8:2::1/64
Device N	GE0/1/0	2001:db8:2::2/64
	GE0/2/0	2001:db8:3::1/64
Device D	GE0/1/0	2001:db8:1::2/64
	GE0/2/0	2001:db8:3::2/64
	GE0/3/0	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 0/1/0
[~DeviceS-GigabitEthernet0/1/0] ipv6 enable
[*DeviceS-GigabitEthernet0/1/0] ipv6 address 2001:db8:1::1/64
[*DeviceS-GigabitEthernet0/1/0] commit
[~DeviceS-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceS-GigabitEthernet0/1/0] isis ipv6 enable 10
[*DeviceS-GigabitEthernet0/1/0] quit
[*DeviceS] interface gigabitethernet 0/2/0
[*DeviceS-GigabitEthernet0/2/0] isis ipv6 enable 10
[*DeviceS-GigabitEthernet0/2/0] commit
[~DeviceS-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceN-GigabitEthernet0/1/0] isis ipv6 enable 10
[*DeviceN-GigabitEthernet0/1/0] quit
[*DeviceN] interface gigabitethernet 0/2/0
[*DeviceN-GigabitEthernet0/2/0] isis ipv6 enable 10
[*DeviceN-GigabitEthernet0/2/0] commit
[~DeviceN-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] isis ipv6 enable 10
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis ipv6 enable 10
[*DeviceD-GigabitEthernet0/2/0] quit
[*DeviceD] interface gigabitethernet 0/3/0
[*DeviceD-GigabitEthernet0/3/0] isis ipv6 enable 10
[*DeviceD-GigabitEthernet0/3/0] commit
[~DeviceD-GigabitEthernet0/3/0] quit
```

Run the **display ipv6 routing-table** command. The command output shows that routers have learned IPv6 routes from each other.

Step 3 Configure the IS-IS cost for each interface.

Configure Device S.

```
[~DeviceS] interface gigabitethernet 0/1/0
[~DeviceS-GigabitEthernet0/1/0] isis ipv6 cost 1 level-2
[*DeviceS-GigabitEthernet0/1/0] quit
[*DeviceS] interface gigabitethernet 0/2/0
[*DeviceS-GigabitEthernet0/2/0] isis ipv6 cost 10 level-2
[*DeviceS-GigabitEthernet0/2/0] commit
[~DeviceS-GigabitEthernet0/2/0] quit
```

Configure Device N.

```
[~DeviceN] interface gigabitethernet 0/1/0
[~DeviceN-GigabitEthernet0/1/0] isis ipv6 cost 10 level-2
[*DeviceN-GigabitEthernet0/1/0] quit
[*DeviceN] interface gigabitethernet 0/2/0
[*DeviceN-GigabitEthernet0/2/0] isis ipv6 cost 10 level-2
[*DeviceN-GigabitEthernet0/2/0] commit
[~DeviceN-GigabitEthernet0/2/0] quit
```

Configure Device D.

```
[~DeviceD] interface gigabitethernet 0/1/0
[~DeviceD-GigabitEthernet0/1/0] isis ipv6 cost 1 level-2
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis ipv6 cost 10 level-2
[*DeviceD-GigabitEthernet0/2/0] commit
[~DeviceD-GigabitEthernet0/2/0] 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 : GE0/1/0
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 : GE0/2/0
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 0/1/0.

```
[~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    : GigabitEthernet0/1/0                  Flags        : D
```

Run the **shutdown** command on GE 0/1/0 of Device D to simulate a fault in the primary link.

```
[~DeviceD] interface gigabitethernet 0/1/0
[~DeviceD-GigabitEthernet0/1/0] shutdown
[*DeviceD-GigabitEthernet0/1/0] 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    : GigabitEthernet0/2/0            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 0/2/0; 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 : GE0/2/0
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): 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
 isis ipv6 enable 10
 isis ipv6 cost 1 level-2
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
 isis ipv6 enable 10
 isis ipv6 cost 10 level-2
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
 isis ipv6 enable 10
 isis ipv6 cost 1 level-2
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::2/64
 isis ipv6 enable 10
 isis ipv6 cost 10 level-2
 #
interface GigabitEthernet0/3/0
 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.

1.1.9.38.9 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 1-88**:

- 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 1-88 Configuring basic IS-IS IPv6 functions

Device Name	Interface	IP Address
Device A	GE 0/1/0	2001:db8:1::2/64
Device B	GE 0/1/0	2001:db8:2::2/64
Device C	GE 0/1/0	2001:db8:1::1/64
	GE0/2/0	2001:db8:2::1/64
	GE 0/3/0	2001:db8:3::1/64
Device D	GE 0/1/0	2001:db8:3::2/64

Device Name	Interface	IP Address
	GE 0/2/0	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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ipv6 enable
[*DeviceA-GigabitEthernet0/1/0] ipv6 address 2001:db8:1::2 64
[*DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] interface gigabitethernet 0/3/0
[*DeviceC-GigabitEthernet0/3/0] isis ipv6 enable 1
[*DeviceC-GigabitEthernet0/3/0] isis circuit-level level-1-2
[*DeviceC-GigabitEthernet0/3/0] commit
[~DeviceC-GigabitEthernet0/3/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis ipv6 enable 1
[*DeviceD-GigabitEthernet0/2/0] commit
[~DeviceD-GigabitEthernet0/2/0] 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              GigabitEthernet0/1/0       FE80::A83E:0:3ED2:1    10          A/-/-
 2001:db8:1::/64   GigabitEthernet0/1/0       Direct                 10          D/L/-
 2001:db8:2::/64   GigabitEthernet0/1/0       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   GigabitEthernet0/1/0   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 GigabitEthernet0/2/0          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 GigabitEthernet0/3/0          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/OL
-------------------------------------------------------------------------------
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* 0x00000017   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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
 isis ipv6 enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::1/64
 isis ipv6 enable 1
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:3::2/64
 isis ipv6 enable 1
#
interface GigabitEthernet0/2/0
 ipv6 enable
 ipv6 address 2001:db8:4::1/64
 isis ipv6 enable 1
#
return
```

1.1.9.38.10 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 1-89**:

- 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 1-89 Configuring IS-IS Auto FRR

Device Name	Interface	IP Address
DeviceA	GE0/1/0	10.0.1.1/24
	GE0/2/0	10.1.1.1/24
DeviceB	GE0/1/0	10.1.1.2/24
	GE0/2/0	10.2.1.1/24
DeviceC	GE0/1/0	10.0.1.2/24
	GE0/2/0	10.3.1.1/24
DeviceD	GE0/1/0	10.3.1.2/24
	GE0/2/0	10.2.1.2/24
	GE0/3/0	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 0/2/0 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/2/0
[*DeviceA-GigabitEthernet0/2/0] isis enable 1
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] isis enable 1
[*DeviceB-GigabitEthernet0/2/0] commit
[~DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable 1
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis enable 1
[*DeviceC-GigabitEthernet0/2/0] commit
```

 Copyright © Huawei Technologies Co., Ltd.

```
[~DeviceC-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] isis enable 1
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis enable 1
[*DeviceD-GigabitEthernet0/2/0] commit
[~DeviceD-GigabitEthernet0/2/0] quit
```

Step 3 Set the cost of Gigabit Ethernet 0/2/0 on Device A to 30, and then check routing information.

Configure the cost of GE 0/2/0 on DeviceA to 30.

```
[~DeviceA] interface gigabitethernet 0/2/0
[~DeviceA-GigabitEthernet0/2/0] isis cost 30
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] 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                      GE0/1/0                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                      GE0/1/0                0x00000010
   (B)10.1.1.2                    GE0/1/0                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 0/1/0
 undo shutdown
 ip address 10.0.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet 0/2/0
 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 0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet 0/2/0
 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 0/1/0
 undo shutdown
 ip address 10.0.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet 0/2/0
 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 0/1/0
 undo shutdown
 ip address 10.3.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet 0/2/0
 undo shutdown
 ip address 10.2.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet 0/3/0
 undo shutdown
 ip address 10.4.1.1 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.11 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 1-90**, 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 1-90 Configuring IS-IS MT

Device Name	Interface	IP Address
Device A	GE 0/1/0	192.168.1.1/24
	GE 0/2/0	2001:db8:1::1/64
Device B	GE 0/1/0	192.168.1.2/24
	GE 0/2/0	192.168.2.1/24
Device C	Loopbak1	2001:db8:3::1/64
	GE 0/1/0	2001:db8:2::2/64
	GE 0/2/0	192.168.2.2/24
Device D	GE 0/1/0	2001:db8:1::2/64
	GE 0/2/0	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 1-90**, 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 1-90**. 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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] isis cost 4
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ip topology red enable
[*DeviceA-GigabitEthernet0/1/0] isis topology red
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
[~DeviceA] interface gigabitethernet 0/2/0
[~DeviceA-GigabitEthernet0/2/0] ip topology red enable
[*DeviceA-GigabitEthernet0/2/0] isis topology red
[*DeviceA-GigabitEthernet0/2/0] ipv6 topology blue enable
[*DeviceA-GigabitEthernet0/2/0] isis ipv6 topology blue
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] 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    GE0/2/0         FE80::D11:0:36D4:1          14         A/-/-
 2001:db8:2::/64    GE0/2/0         Direct                   14         D/L/-
 2001:db8:1::/64    GE0/1/0         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 0/2/0.

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    GE0/1/0         FE80::200:5EFF:FE01:100     13         A/-/-
 2001:db8:2::/64    GE0/2/0         Direct                   14         D/L/-
 2001:db8:1::/64    GE0/1/0         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 GE0/1/0 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    GE0/1/0        192.168.1.2     A/-/-/-
 192.168.1.0/24     4       NULL    GE0/1/0        Direct          D/-/L/-
 IPV6 Dest.          ExitInterface   NextHop                    Cost       Flags
-------------------------------------------------------------------------------
 2001:db8:2::/64     GE0/1/0         FE80::2E0:A9FF:FE47:8302     24         A/-/-
 2001:db8:1::/64     GE0/2/0         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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
 isis enable 1
 ip topology red enable
 isis topology red
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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
```

1.1.9.38.12 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 1-91**, 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 1-91 Routing loop detection for routes imported from OSPF to IS-IS

NOTE

In this example, interface1, interface2, and interface3 represent GE0/1/0, GE0/2/0, and GE0/3/0, 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. Disable routing loop detection for routes imported into IS-IS on each device. This function is enabled by default.
2. Configure IP addresses for interfaces on each device to implement area connectivity.
3. Enable IS-IS and OSPF, and configure basic IS-IS and OSPF functions.
4. Configure route import to construct a routing loop.
5. View routing entries to check whether a routing loop occurs.
6. Enable routing loop detection and view routing entries to check whether the routing loop is eliminated.

Procedure

Step 1 Disable routing loop detection for routes imported into IS-IS on each device. This function is enabled by default.

Disable routing loop detection. DeviceA is used as an example.

```
<DeviceA> system-view
[~DeviceA] route loop-detect isis disable
[*DeviceA] commit
```

The configurations of other devices are similar to those of DeviceA. For configuration details, see **Configuration Files** in this section.

Step 2 Configure an IP address for each interface.

DeviceA, DeviceB, DeviceC, and DeviceD are used as examples.

Configure DeviceA.

```
<DeviceA> system-view
[~DeviceA] interface gigabitethernet 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ip address 10.12.1.1 24
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/2/0
[*DeviceA-GigabitEthernet0/2/0] ip address 10.16.1.1 24
[*DeviceA-GigabitEthernet0/2/0] quit
[*DeviceA] commit
```

Configure DeviceB.

```
<DeviceB> system-view
[~DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ip address 10.12.1.2 24
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] ip address 10.26.1.2 24
[*DeviceB-GigabitEthernet0/2/0] quit
[*DeviceB] interface gigabitethernet 0/3/0
[*DeviceB-GigabitEthernet0/3/0] ip address 10.23.1.2 24
[*DeviceB-GigabitEthernet0/3/0] quit
[*DeviceB] commit
```

Configure DeviceC.

```
<DeviceC> system-view
[~DeviceC] interface gigabitethernet 0/1/0
[*DeviceC-GigabitEthernet0/1/0] ip address 10.23.1.3 24
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ip address 10.35.1.3 24
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] interface gigabitethernet 0/3/0
[*DeviceC-GigabitEthernet0/3/0] ip address 10.34.1.3 24
[*DeviceC-GigabitEthernet0/3/0] 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] 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 3 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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] isis enable 2
[*DeviceA-GigabitEthernet0/1/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet0/1/0] isis cost 10
[*DeviceA-GigabitEthernet0/1/0] quit
[*DeviceA] interface gigabitethernet 0/2/0
[*DeviceA-GigabitEthernet0/2/0] isis enable 2
```

```
[*DeviceA-GigabitEthernet0/2/0] isis circuit-type p2p
[*DeviceA-GigabitEthernet0/2/0] isis cost 10
[*DeviceA-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] isis enable 1
[*DeviceD-GigabitEthernet0/1/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet0/1/0] isis cost 10
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis enable 1
[*DeviceD-GigabitEthernet0/2/0] isis circuit-type p2p
[*DeviceD-GigabitEthernet0/2/0] isis cost 10
[*DeviceD-GigabitEthernet0/2/0] 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] network 10.23.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 10.26.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] quit
[*DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ospf enable 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/1/0] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] ospf enable 1 area 0.0.0.0
[*DeviceB-GigabitEthernet0/2/0] 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 4 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
[*DeviceC-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 5 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 (GigabitEthernet0/3/0)'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 (GigabitEthernet0/2/0)'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  GE0/3/0          0000000011        Up   29s      L1L2     --
1111.0000.0001  GE0/2/0          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       GigabitEthernet0/3/0
```

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 (GigabitEthernet0/3/0)'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 (GigabitEthernet0/2/0)'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  GE0/2/0          0000000011        Up   28s      L1L2     --
1111.0000.0001  GE0/1/0          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       GigabitEthernet0/2/0
```

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 6 Enable routing loop detection on each device.

Configure DeviceA, which is used as an example.

```
[~DeviceA] undo route loop-detect isis disable
[*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 7 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       GigabitEthernet0/3/0
```

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       GigabitEthernet0/2/0
```

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
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.12.1.1 255.255.255.0
 isis enable 2
 isis circuit-type p2p
 isis cost 10
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.16.1.1 255.255.255.0
 isis enable 2
 isis circuit-type p2p
 isis cost 10
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
isis 2
 cost-style wide
 network-entity 10.2222.0000.0001.00
 import-route ospf 1
#
```

```
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.12.1.2 255.255.255.0
 isis enable 2
 isis circuit-type p2p
 isis cost 10
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/3/0
 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
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 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 GigabitEthernet0/3/0
 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
#
interface LoopBack1
 ip address 10.1.1.1 255.255.255.255
 isis enable 1
 isis cost 10
#
return
```

- DeviceE configuration file

```
#
sysname DeviceE
#
isis 1
 cost-style wide
 network-entity 10.5555.0000.0001.00
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.56.1.5 255.255.255.0
 ospf timer hello 1
 ospf enable 1 area 0.0.0.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.35.1.5 255.255.255.0
 ospf timer hello 1
 ospf enable 1 area 0.0.0.0
 isis enable 1
 isis circuit-type p2p
 isis cost 10
#
interface GigabitEthernet0/3/0
 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 0.0.0.0
#
route loop-detect ospf enable
#
return
```

- DeviceF configuration file

```
#
sysname DeviceF
#
isis 2
 cost-style wide
 network-entity 10.6666.0000.0001.00
 #
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.16.1.6 255.255.255.0
 isis enable 2
 isis circuit-type p2p
 isis cost 10
#
```

```
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/3/0
 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
```

1.1.9.38.13 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 1-92 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 1-92 IS-IS route summarization networking

Device Name	Interface	IP Address
Device A	GE 0/2/0	172.17.1.1/24
Device B	GE 0/1/0	172.16.4.2/24
	GE 0/2/0	172.17.1.2/24
Device C	GE 0/1/0	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 0/2/0
[*DeviceA-GigabitEthernet0/2/0] isis enable 1
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] quit
```

Configure Device B.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] isis enable 1
[*DeviceB-GigabitEthernet0/2/0] quit
[*DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable 1
[*DeviceC-GigabitEthernet0/1/0] commit
[~DeviceC-GigabitEthernet0/1/0] quit
```

The configurations of GE 0/2/0, GE 0/3/0, and GE 0/1/1 are similar to those of GE 0/1/0. For configuration details, see **Configuration Files** in this section.

Display the IS-IS routing information on Device A.

```
[~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    GE0/2/0        172.17.1.2      A/-/L/-
 172.16.2.0/24        30         NULL    GE0/2/0        172.17.1.2      A/-/L/-
 172.16.3.0/24        30         NULL    GE0/2/0        172.17.1.2      A/-/L/-
 172.16.4.0/24        20         NULL    GE0/2/0        172.17.1.2      A/-/L/-
 172.17.1.0/24        10         NULL    GE0/2/0        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    GE0/2/0        172.17.1.2      A/-/L/-
 172.17.1.0/24         10         NULL    GE0/2/0        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 GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ip address 172.17.1.2 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
```

```
 undo shutdown
 ip address 172.16.4.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.2.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 172.16.3.1 255.255.255.0
 isis enable 1
#
return
```

1.1.9.38.14 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 1-93 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 1-93 Local IS-IS MT networking

Device Name	Interface	IP Address
DeviceA	GE0/1/0	172.16.1.1/24
	GE0/2/0	10.0.0.1/24
DeviceB	GE0/1/0	10.0.0.2/24
	GE0/2/0	10.0.1.2/24
DeviceC	GE0/1/0	10.0.1.1/24
	GE0/2/0	10.0.2.2/24
DeviceD	GE0/1/0	10.0.3.1/24
	GE0/2/0	10.0.2.1/24
DeviceE	GE0/1/0	10.0.3.3/24
	GE0/2/0	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 1-93**), 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 1-93**, 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 GigabitEthernet0/1/0 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 0/2/0
[*DeviceA-GigabitEthernet0/2/0] pim sm
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] quit
```

Enable IGMP on the interface through which DeviceA is connected to hosts.

```
[~DeviceA] interface gigabitethernet 0/1/0
[~DeviceA-GigabitEthernet0/1/0] igmp enable
[*DeviceA-GigabitEthernet0/1/0] igmp version 3
[*DeviceA-GigabitEthernet0/1/0] 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 gigabitethernet 0/1/0
[*DeviceD-pim] c-rp gigabitethernet 0/1/0
[*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: GigabitEthernet0/2/0
    List of 1 downstream interface
      1:  GigabitEthernet0/1/0
```

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 0/2/0
[*DeviceB-GigabitEthernet0/2/0] mpls
[*DeviceB-GigabitEthernet0/2/0] mpls te
[*DeviceB-GigabitEthernet0/2/0] mpls rsvp-te
[*DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] mpls
[*DeviceC-GigabitEthernet0/1/0] mpls te
[*DeviceC-GigabitEthernet0/1/0] mpls rsvp-te
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] mpls
[*DeviceC-GigabitEthernet0/2/0] mpls te
[*DeviceC-GigabitEthernet0/2/0] mpls rsvp-te
[*DeviceC-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] mpls
[*DeviceD-GigabitEthernet0/1/0] mpls te
[*DeviceD-GigabitEthernet0/1/0] mpls rsvp-te
[*DeviceD-GigabitEthernet0/1/0] 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 igp shortcut isis
[*DeviceB-Tunnel10] mpls te igp 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    GE0/2/0         10.0.1.1         A/-/-/-
 172.16.1.0/24        20         NULL    GE0/1/0         10.0.0.1         A/-/-/-
 2.2.2.2/32           0          NULL    Loop0           Direct           D/-/L/-
 192.168.3.0/24       25         NULL    Tun0/1/0        2.2.2.2          A/S/-/-
 5.5.5.5/32           15         NULL    Tun0/1/0        2.2.2.2          A/S/-/-
 10.0.0.0/24          10         NULL    GE0/1/0         Direct           D/-/L/-
 10.0.1.0/24          10         NULL    GE0/2/0         Direct           D/-/L/-
 4.4.4.4/32           5          NULL    Tun0/1/0        2.2.2.2          A/S/-/-
 10.0.2.0/24          15         NULL    Tun0/1/0        2.2.2.2          A/S/-/-
 10.0.3.0/24          15         NULL    Tun0/1/0        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: GigabitEthernet0/2/0
      List of 1 downstream interface
          1:  GigabitEthernet0/1/0
```

Display the MIGP routing table on DeviceB.

```
[~DeviceB] display migp 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
      4.4.4.4/32    ISIS   15   20            10.0.1.1        GE0/2/0
      5.5.5.5/32    ISIS   15   30            10.0.1.1        GE0/2/0
     10.0.2.0/24    ISIS   15   20            10.0.1.1        GE0/2/0
     10.0.3.0/24    ISIS   15   30            10.0.1.1        GE0/2/0
  192.168.3.0/24    ISIS   15   40            10.0.1.1        GE0/2/0
```

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 GigabitEthernet0/2/0
 ip address 10.0.0.1 255.255.255.0
 isis enable 1
 pim sm
#
interface GigabitEthernet0/1/0
 ip address 172.16.1.1 255.255.255.0
 isis enable 1
 igmp enable
#
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 GigabitEthernet0/1/0
 ip address 10.0.0.2 255.255.255.0
 isis enable 1
 pim sm
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 ip address 10.0.1.1 255.255.255.0
 isis enable 1
 pim sm
 mpls
 mpls te
 mpls rsvp-te
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 ip address 10.0.3.1 255.255.255.0
 isis enable 1
 pim sm
#
interface GigabitEthernet0/2/0
 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
#
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 GigabitEthernet0/1/0
 ip address 10.0.3.3 255.255.255.0
 isis enable 1
 pim sm
#
interface GigabitEthernet0/2/0
 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
```

1.1.9.38.15 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 1-94**:

- Device A and Device B belong to the same AS, and an IS-IS neighbor relationship is established between the two devices.

Figure 1-94 Networking diagram for configuring IS-IS route recursion to IPv6 next hops

Device Name	Interface	IP Address
DeviceA	GE0/1/0	2001:DB8:1::1/64
DeviceB	Loopback0	2.2.2.2/32
	GE0/1/0	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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] ipv6 enable
[*DeviceA-GigabitEthernet0/1/0] ipv6 address 2001:DB8:1::1 64
[*DeviceA-GigabitEthernet0/1/0] isis enable 1
[*DeviceA-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] quit
```

Configure DeviceB.

```
[~DeviceB] isis 1
[*DeviceB-isis-1] network-entity 10.0000.0000.0002.00
[*DeviceB-isis-1] quit
[*DeviceB] interface gigabitethernet 0/1/0
[*DeviceB-GigabitEthernet0/1/0] ipv6 enable
[*DeviceB-GigabitEthernet0/1/0] ipv6 address 2001:DB8:1::2 64
[*DeviceB-GigabitEthernet0/1/0] isis enable 1
[*DeviceB-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] quit
```

Step 2 Configure IS-IS route recursion to IPv6 next hops.

Configure DeviceA.

```
[~DeviceA] isis 1
[*DeviceB-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    GigabitEthernet0/1/0          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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

1.1.10 BGP Configuration

Border Gateway Protocol (BGP) is applicable to complicated large-scale networks and used to transmit routing information between ASs.

1.1.10.1 BGP Overview

The Border Gateway Protocol (BGP) advertises and maintains a large number of routes between autonomous systems (ASs).

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 that prevents route flapping, which effectively enhances Internet stability.
- BGP can be easily extended.

Purpose

BGP transmits route information between ASs. It, however, is not required in all scenarios.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-95 BGP networking

BGP is required in the following scenarios:

- On the network shown in **Figure 1-95**, 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. Routers, 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 NE40E-M2 series Feature Description - VPN*.
- Users need to transmit multicast routes and construct a multicast topology. For details, see the *HUAWEI NE40E-M2 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.

1.1.10.2 Configuration Precautions for BGP

Feature Requirements

Table 1-17 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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
BGP routes cannot be recursed to SRv6 BE LSPs. After recursion, the routes are inactive.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
(1) Configuration restrictions: A maximum of 100 regions can be configured in one VS. A maximum of 100 AS numbers (either 4-byte or 2-byte) can be configured in one region. A maximum of 100 regional confederations can be configured in one VS. A maximum of 100 regions can be configured in one regional confederation. The same AS cannot be added to different regions. The display format is not affected. For example, 1.0 and 65536 indicate the same AS. The same region cannot be added to different regional confederations. The regions to be added to a regional confederation must exist. If a region has been added to a regional confederation, deleting the region will delete the region's information in the regional confederation. (2) Regional authentication depends on the trustworthiness of the original AS in each route. In actual application, the RPKI ROA function must also be deployed to ensure the correctness of the original AS. (3) The regional authentication configuration is a global configuration. Currently, the scenario where the public and private network ASs overlap is not considered. (4) Regional authentication can be enabled or disabled only in the address family view, not for a single peer. Currently, the following address families support regional authentication: IPv4 unicast address family, IPv6 unicast address family, VPN instance IPv4 address family, VPN instance IPv6 address family, BGP multi-instance VPN instance IPv4 address family, IPv4 unicast label address family, and IPv6 unicast label address family (5) Regional authentication must be configured on a border router that is directly connected to a router in an external region. Risky internal routes that already exist cannot be identified. (6) When both an import policy (for example, to modify AS numbers) and regional authentication are configured, regional	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

Feature Requirements	Series	Models
authentication is performed before the import policy takes effect (after ROA). (7) Regional authentication affects the route learning performance of EBGP peers (the learning performance cannot decrease by more than 5% after regional authentication is configured). Regional authentication does not affect the convergence performance, RR reflection performance, or IBGP peers' route learning performance.		
The routing policy configured using the apply cost-type med-inherit-aigp command takes effect on the IPv4 and IPv6 private networks. The AIGP attribute of the private network route on the PE is advertised to the CE through the MED attribute. The policy applied to other address families does not take effect.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In a public network IP over SRv6 TE Policy scenario, mirror-SID-based egress protection and service FRR-based egress protection are not supported.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In the SR-MPLS TE Policy address family, IP prefix list, ACL, and RD filters cannot be used to filter routes based on NLRI. If the route-policy configured for a peer in the SR-MPLS TE Policy address family contains the if-match ip-prefix, if-match acl, or if-match rd-filter configuration, all the routes of the peer match the route-policy. If the route-policy configured for a peer in the SR-MPLS TE Policy address family contains the if-match as-path or if-match cost configuration, only the routes that meet the filtering rules match the route-policy. Properly plan route-policies. Do not use unsupported matching modes.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
Load balancing is not supported between MPLS and SRv6 tunnels.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
BGP IPv4 public network unicast routes, BGP IPv4 VPN remote cross routes, BGP IPv4 public network labeled routes (in 6PE networking), and BGP IPv6 VPN remote cross routes can recurse to SR-MPLS TE Policy tunnels based on next hop+color. Other routes do not support such recursion. Properly plan the tunnel policy and tunnel selector.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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 can recurse to SR-MPLS TE Policy tunnels. If multiple color extended community attributes are carried, only the maximum color value is used for recursion to SR-MPLS TE Policy tunnels. In addition, the value of the CO flag (refer to draft-ietf-idr-segment-routing-te-policy) in each received route is considered 00, regardless of its actual value. Properly plan the color extended community attribute to be added to routes on the egress.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
Allocation of DX4 SIDs for BGP public network IPv4 routes by the next hop is mutually exclusive with the Add-path function. If Add-path has been enabled before a DX4 SID is applied for, no DX4 SID can be obtained. If a DX4 SID has been obtained before the Add-path function is enabled, the allocated DX4 SID is released.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
IPv4 and IPv6 neighbor functions cannot be enabled at the same time in the VPNv4 address family.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
When the apply extcommunity soo command is run in the XPL policy and route-policy, only the apply extcommunity soo {<source-of-origin> &<1-16>} additive command takes effect.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
Scenario 1: The as-number command is run in the BGP-VPN instance IPv4 or IPv6 address family view to configure a VPN AS number, and the bgp yang-mode enable command is run to enable the YANG management mode for the BGP VPN instance. Restrictions: 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, configurations in the configuration file fail to be pasted. To solve this problem, configure a VPN AS number using the as-number command first and then run the other commands in the configuration file. 2. Do not configure BGP peer relationships 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. Do not configure BGP-VPN IPv4 peer relationships in the BGP-VPN instance IPv6 address family view. 3. Before deleting the VPN AS number, delete all the peers and peer groups in the BGP-VPN instance IPv4 or IPv6 address family as well as all the peers in the BGP VPN instance view. 4. Before running the undo bgp yang-mode enable command, delete the VPN AS number. Scenario 2: A VPN AS number is configured in both the BGP-VPN instance IPv4 address family view and BGP-VPN instance IPv6 address family view, and the YANG management mode is enabled for the BGP VPN instance and then disabled. Restrictions: 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, configurations in the configuration file fail to be pasted. To solve this problem, run the peer as-number command in both the BGP-VPN instance IPv4 address family view and BGP-VPN instance IPv6 address family view to configure a new peer or peer group. However, do not configure a new peer in the BGP-VPN instance view.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
2. Do not configure BGP peer relationships 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. Do not configure BGP-VPN IPv4 peer relationships in the BGP-VPN instance IPv6 address family view. 3. Before deleting the VPN AS number, delete all the peers and peer groups in the BGP-VPN instance IPv4 or IPv6 address family as well as all the peers in the BGP VPN instance view.		
Pay attention to the following points when using BGP EPE: 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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
If IPv6 routes received from IPv6 peers recurse to 6PE routes, the Pop-Go forwarding mode is not supported.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
The original community values in all BGP public network routes on a device must be different from the configured community values that indicate peer roles. Otherwise, the original community attributes will be replaced, or routes will be incorrectly filtered. Users need to ensure that the community values are unique during the overall network planning.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
In BGP EPEv6 scenarios, only End/End.X SIDs support compression.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
UNRs do not support POPGO.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

 Copyright © Huawei Technologies Co., Ltd.

Feature Requirements	Series	Models
1. If a device on the loop 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. Workaround: Ensure that loop detection is enabled on each device on the loop path and the number of devices on the loop path does not exceed four. Impact: The loop detection function does not take effect. 2. If the local preference or MED of a looped route is reduced and then changed to another value using a routing policy, the priority of the looped route cannot be reduced. As a result, the loop cannot be eliminated. Workaround: Do not use a routing policy to modify the local-preference or MED of the routes that form a loop. Impact: The loop detection function does not take effect. 3. If the priority of the original route is the lowest, the priority cannot be reduced to prevent loops. Preventive measure: none Impact: The loop detection function does not take effect. 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. Preventive measure: none 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. 5. The maximum number of looped routes that can be recorded is controlled by the PAF file. Excess loop routes are not recorded. As a result, loops may fail to be removed for some routes. Preventive measure: none Impact: The loop detection function does not take effect.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

Feature Requirements	Series	Models
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, looped routes may be misjudged. Preventive measure: none Impact: Looped routes are incorrectly determined. 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. Workaround: If both source routes and looped routes exist on a device, remove the loop first and then delete the routes. Impact: The looped route flaps on the loop path, and the route cannot be withdrawn after a loop occurs.		
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
ROA verification is not performed on the routes received from IBGP peers and locally imported routes.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.10.3 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.

1.1.10.3.1 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.

 Copyright © Huawei Technologies Co., Ltd.

NOTE

By default, BGP automatically selects a router ID in the system view as its router ID. If the selected router ID is the IP address of a physical interface, route flapping occurs when the IP address changes. 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

1.1.10.3.2 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 not be established.

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. 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

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 not be established.

e. (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.

f. (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.

g. (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.

h. (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.

i. Run **commit**

The configuration is committed.

----End

1.1.10.3.3 Configuring BGP to Import Routes

BGP can import the routes from other routing protocols. When BGP needs to import routes from a dynamic routing protocol, you need to specify the process ID of the protocol.

Context

BGP itself cannot discover routes. Therefore, it needs to import routes from other protocols, such as IGP or static routes and adds the routes to the BGP routing table so that these imported routes can be transmitted within an AS or between ASs.

BGP imports routes in the following modes:

- Import mode: BGP imports routes by protocol type, such as RIP, OSPF, IS-IS, static routes, and 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, specify **route-policy** *route-policy-name* or **route-filter** *route-filter-name* in the command.

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 but tunnel stitching. As a result, traffic is diverted to the peer MASG, which prolongs the traffic switching time, and egress protection fails. To address this problem, specify the **non-relay-tunnel** parameter to prevent the routes imported by BGP from being recursed to tunnels.

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.

To import default routes, run both the **default-route imported** command and the **import-route** command.

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 flexibly control the routes to be imported, specify the **route-policy** *route-policy-name* parameter.

To use a route-filter to flexibly control 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 but tunnel stitching. As a result, traffic is diverted to the peer MASG, which prolongs the traffic switching time, and egress protection fails. To address this problem, specify the **non-relay-tunnel** parameter to prevent the routes imported by BGP from being recursed to tunnels.

NOTE

- A route with the destination address and mask specified in the **network** command can be imported only when the specified items exactly match a routing entry in the local IP routing table.
- When running the **undo network** command to delete the existing configuration, you need to specify the correct mask.

e. Run **commit**

The configuration is committed.

----End

1.1.10.3.4 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 IDs.
- 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

1.1.10.4 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. To use integral 4-byte AS numbers, perform this configuration task to display 4-byte AS numbers as integers.

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 must 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 AS numbers. Otherwise, routes cannot match the export or import policy, and a network fault occurs.

- If integral 4-byte AS numbers are configured, you must change 4-byte AS numbers in AS_Path regular expressions and extended community attribute filters to integral 4-byte AS numbers.
- 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.

1.1.10.5 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**.

1.1.10.5.1 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 be 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

However, 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

1.1.10.5.2 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 the **peer** { *group-name* | *ipv4-address* } **preferred-value** *preferredvalue* command to set a PrefVal 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

1.1.10.5.3 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

1.1.10.5.4 Configuring MED Attributes for BGP Routes

The MED attribute equals the metric used by an IGP. After the MED attributes of routes are set, an EBGP peer selects the route with the smallest MED value for the traffic that enters an AS if the other attributes of the routes are the same.

Context

If the Router running BGP obtains multiple routes from different EBGP peers and these routes have different next hops but the same destination, the BGP device selects the route with the smallest MED value.

Procedure

- Set the default MED value on a 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 **default med** *med*

     A default MED value is configured.

     NOTE

     The **default med** command is valid only for routes imported using the **import-route** command and summary BGP routes on the local Router.
  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 **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** [**igp-multiplier** *igp-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**

The device is configured to remove the MED attribute from the imported routes that are locally leaked and are to be advertised to the peer if an export route-policy in which the **apply cost-type** command is run is configured for the peer.

After an export route-policy in which the **apply cost-type** command is run is applied to the routes to be advertised to a peer, you can run the **local-cross-routing non-med** command to remove the MED attribute from the imported routes that are locally leaked and are to be advertised to the peer.

d. Run **commit**

The configuration is committed.

----End

1.1.10.5.5 Configuring the Next_Hop Attribute

Configuring the Next_Hop attribute allows for flexible control of BGP route selection.

Procedure

- Configure a device to change the next hop address of a route when the device advertises the route to an IBGP peer.

  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.

  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.

d. Run the **peer** { *ipv4-address* | *group-name* } **next-hop-invariable** command to configure the device not to change the next hop address of each imported IGP route when advertising the route, or run the **peer** { *ipv4-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 public network static route is invalid, the next hop of the public network static route belongs to a VPN instance, 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 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 vpnv4** [**unicast**]

The BGP-VPNv4 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 addresses of routes before the device advertises the routes 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 addresses of routes before the RR advertises 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.

On the network shown in **Figure 1-96**, a BGP LSP is established between PE1 and PE2. VPNv4 routes are exchanged between PE1 and RR1, between RR1 and RR2, and between RR2 and PE2 through BGP VPNv4 peer relationships.

Figure 1-96 Inter-AS VPN Option C networking with RRs deployed

Assume that PE1 needs to advertise a VPNv4 route to PE2. The route will be advertised through the following process:

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. Upon receipt, RR2 advertises the route to PE2 through the IBGP peer relationship. By default, the Router changes the next hop addresses of labeled routes received from an EBGP peer before advertising the routes to an IBGP peer. Therefore, RR2 changes the next hop address of the received route to its own address before advertising it 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 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 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 1-97**, DeviceA establishes EBGP peer relationships with DeviceB and DeviceC, but DeviceB and DeviceC do not establish a peer relationship with each other.

Figure 1-97 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

1.1.10.5.6 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 number of repeated local AS numbers.

  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 is used to hide the actual AS number. EBGP peers in other ASs can learn only this fake AS number. The peers in other ASs use 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 the **system-view** command to enter the system view.

b. Run the **bgp** *as-number* command to enter the BGP view.
c. Run the **ipv4-family** { **vpn-instance** *vpn-instance-name* | **unicast** } command to enter the BGP-VPN instance IPv4 address family or BGP-IPv4 unicast address family view.
d. Run the **peer** { *ipv4-address* | *group-name* } **substitute-as** command to enable AS number substitution on the device.
e. Run the **commit** command to commit the configuration.

- 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 the **peer** { *ipv4-address* | *group-name* } **public-as-only** [**force** [**replace**] [**include-peer-as**] | **limited** [**replace**] [**include-peer-as**]] command to configure the AS_Path attribute in the BGP Update messages to be sent to carry only public AS numbers. Alternatively, 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

1.1.10.5.7 Configuring AIGP Attributes for Routes

The Accumulated Interior Gateway Protocol Metric (AIGP) attribute allows devices in an AIGP administrative domain to use the optimal routes to forward data.

Context

An AIGP administrative domain is a set of autonomous systems (ASs) in a common administrative domain.

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 runs several contiguous 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 the **ipv4-family unicast** command to enter the BGP-IPv4 unicast address family view, the **ipv4-family labeled-unicast** command to enter the BGP-labeled address family view, or the **ipv4-family vpnv4** command to enter the BGP-VPNv4 address family view.

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 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 VPN routes to participate in route selection using the AIGP attribute of the BGP LSP through which they are transmitted.

1. Run **system-view**

   The system view is displayed.

2. Run **bgp** *as-number*

   The BGP view is displayed.

3. Run **ipv4-family vpn-instance** *vpn-instance-name*

   The BGP-VPN instance IPv4 address family view is displayed.

4. Run **bestroute nexthop-resolved aigp**

   VPN routes are allowed to participate in route selection using the AIGP attribute of the BGP LSP through which they are transmitted.

5. Run **commit**

   The configuration is committed.

Step 3 (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

1.1.10.5.8 Configuring Attr_Set Attribute Encapsulation or Parsing

Configuring Attr_Set attribute encapsulation or parsing ensures that the attributes of the routes sent by CEs 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 Run **commit**

The configuration is committed.

----End

1.1.10.5.9 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

1.1.10.6 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**.

1.1.10.6.1 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 NE40E-M2 seriesUniversal Service Router Configuration Guide - IP Services.

  An ACL can be used as a filtering condition of a route-policy or used in the **filter-policy** { *acl-number* | **acl-name** *acl-name* } **export** [**direct** | **isis** *process-id* | **ospf** *process-id* | **rip** *process-id* | **static**] or **peer** { *group-name* | *ipv4-address* } **filter-policy** { *acl-number* | **acl-name** *acl-name* } **export** command.

- 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 rule of a route-policy or used in the **filter-policy ip-prefix** *ip-prefix-name* **export** [**direct** | **isis** *process-id* | **ospf** *process-id* | **rip** *process-id* | **static**] or **peer** { *group-name* | *ipv4-address* } **ip-prefix** *ip-prefix-name* **export** command.

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 NE40E 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.

- Metacharacter: defines matching rules.
- Value: defines a matching object.

Table 1-18 Description of metacharacters

Special Character	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. ● 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. ● 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 only. **NOTE** ^$ matches an empty string (empty AS_Path) and is usually used to match routes in the local AS.

 Copyright © Huawei Technologies Co., Ltd.

Special Character	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 ($).	• ^65001_ matches the AS_Paths that begin with 65001 followed by a sign. Specifically, ^65001_ matches AS_Paths with 65001 as the leftmost AS number (the number of the last AS through which a route passes), that is, the routes sent by peers in AS 65001. • _65001_ matches the strings (AS_Paths) that contain 65001, that is, the routes that pass through AS 65001. • _65001$ 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.

 Copyright © Huawei Technologies Co., Ltd.

Special Character	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: ● \(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

Special Character	Function	Example
		in an AS_Confed_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.	65* matches the AS_Paths that begin with 6 and contain zero or multiple 5s. ● 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.	65+ matches the AS_Paths that begin with 6 and contain one or multiple 5s. ● 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.	65? matches the AS_Paths that begin with 6 and contain zero or one 5. ● 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.

Special Character	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 NE40E-M2 seriesUniversal Service 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 basic community filter, run the **ip community-filter basic** *comm-filter-name* [**index** *index-number*]{ **permit** | **deny** }

[*community-number* | *aa:nn* | **internet** | **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.

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 extcommunity 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** { *as-number:nn* | *4as-number:nn* | *ipv4-address:nn* } } &<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 NE40E, 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 configure an AS_Path-based matching rule to filter BGP routes, run the **if-match as-path-filter** *as-path-filter-number* &<1-16> command.
- Match BGP routes against a community filter.
  - **if-match community-filter** { *basic-comm-filter-num* [**whole-match**] | *adv-comm-filter-num* } * &<1-16>
  - **if-match community-filter** *comm-filter-name* [**whole-match**]
  - **if-match community-filter** { *adv-comm-filter-num* **sort-match** } * &<1-16>
  - **if-match community-filter** *comm-filter-name* **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 configure an SoO extended community attribute-based matching rule to filter 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-number-plain* | *as-number-dot* } &<1-128>{ **additive** | **overwrite** | **delete** } or **apply as-path** *asValues* { **additive** | **overwrite** | **delete** } command.
- To delete a specified BGP community attribute from matched routes, run the **apply comm-filter** { *comm-filter-number* | *comm-filter-name* } **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 NE40E-M2 seriesUniversal Service Router Command Reference.
- To delete community attributes from a BGP route, run the **apply community none** command.
- To set community attributes for BGP routes, run the **apply community** { { *community-number* | *aa:nn* } &<1-32> | **internet** | **no-advertise** | **no-export** | **no-export-subconfed** } * [**additive**] or **apply community community-list** *community-list-name* command.

  NOTE

  A BGP community list must be configured using the **ip community-list** command and community attributes must be 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** { *as-number-plain* | *as-number-dot* } | **igp** | **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

1.1.10.6.2 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:

  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 routes to be advertised globally.

- To filter routes based on a basic ACL, perform the following steps:
  1) 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.
  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 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 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 **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** { *peerIpv4Addr* | *peerIpv6Addr* } **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

1.1.10.6.3 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:

       1) Run the **filter-policy** { *acl-number* | **acl-name** *acl-name* } **import** command to filter the routes to be received.
       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 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* | *peerIpv6Addr* } **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* | *peerIpv6Addr* } **route-policy** *route-policy-name* **import** command.

NOTE

A route-policy specified in the **peer route-policy import** command does not support the use of an interface as a matching condition. That is, the **if-match interface** command cannot be used in the route-policy.

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 limit specified.

- If **alert-only** is set, the device does not terminate the connection but does not accept any subsequent 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.
- 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

1.1.10.6.4 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** | **vpnv4**] { **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

1.1.10.6.5 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**] | *advanced-community-filter-number* } 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

1.1.10.7 Configuring BGP XPL

1.1.10.7.1 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:

  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* **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.

1.1.10.7.2 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.

1.1.10.8 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 a 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.
  d. 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 after the **aggregate 10.1.1.1 16** command is run.

e. Run **commit**

The configuration is committed.

----End

Checking the Configurations

After route summarization is configured, you can check whether the configuration is correct.

- Run the **display bgp routing-table** [*network* [*mask* | *mask-length*]] command to check information about BGP summary routes.

1.1.10.9 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 1-98**, 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 1-98 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 BGP-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 configuring BGP to generate a summary default route, verify the configuration.

- Run the **display bgp routing-table** [*network* [*mask* | *mask-length*]] command to check information about the summary default route.

1.1.10.10 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**.

1.1.10.10.1 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* | *ipv6-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** { *ipv4-address* | *ipv6-address* | *group-name* } **description** *description-text*

A description is configured for the peer group.

You can simplify network management by configuring the descriptions of peers.

Step 6 Run **commit**

The configuration is committed.

----End

1.1.10.10.2 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.

NOTE

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

1.1.10.10.3 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* | *ipv6-address* } **as-number** *as-number*

A peer is created, and an AS number is set for this peer.

Step 5 Run **peer** { *ipv4-address* | *ipv6-address* } **group** *group-name*

A peer is added to the peer group.

NOTE

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 of peers.

Step 7 Run **commit**

The configuration is committed.

----End

1.1.10.10.4 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

1.1.10.11 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 x (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 1-99 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 1-99 Typical networking with hierarchical RRs

Pre-configuration Tasks

Before configuring a BGP route reflector, **configure basic BGP functions**.

1.1.10.11.1 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** { *ipv4-address* | *group-name* } **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

1.1.10.11.2 (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

1.1.10.11.3 (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

1.1.10.11.4 (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 no 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.

NOTE

The **routing-table rib-only** and **active-route-advertise** commands are mutually exclusive.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.10.11.5 (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 following configurations can take effect:

- **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 (cost) 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

1.1.10.11.6 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

1.1.10.12 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.

NOTE

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.

1.1.10.13 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**.

1.1.10.13.1 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 **1.1.12.10.2 (Optional) Configuring an if-match Clause**.
  d. Configure community or extended community attributes for BGP routes.
     - 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 **1.1.13.8.2 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

1.1.10.13.2 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 either of the following commands as required to advertise community attributes to a specified peer or peer group.

- To enable the device to advertise a standard community attribute to a specified peer or peer group, run the **peer** { *ipv4-address* | *group-name* } **advertise-community** command.
- To enable the device to advertise an extended community attribute to a specified peer or peer group, run the **peer** { *ipv4-address* | *group-name* } **advertise-ext-community** command.

  After the **peer advertise-ext-community** command is run, BGP advertises the routes with extended community attributes to the specified peer or peer group. 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 on 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. 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 either of the following commands as required to advertise community attributes to a specified peer or peer group.

- To enable the device to advertise a standard community attribute to a specified peer or peer group, run the **peer** { *ipv4-address* | *group-name* } **advertise-community** command.
- To enable the device to advertise an extended community attribute to a specified peer or peer group, run the **peer** { *ipv4-address* | *group-name* } **advertise-ext-community** command.

  After the **peer advertise-ext-community** command is run, BGP advertises the routes with extended community attributes to the specified peer or peer group. If the peer or peer group only needs to accept the routes but not the extended community attributes, you can run the **peer discard-ext-community** command on the peer or peer group to discard the extended community attributes from the received routing information.

  To enable the device to advertise the Link Bandwidth extended community attribute to a specified EBGP peer, run the **peer advertise ebgp link-bandwidth** command.

  To enable the device to convert the Link Bandwidth extended community attribute (optional non-transitive) carried in BGP routes to an optional transitive attribute before advertising the routes to other peers, run the **peer advertise link-bandwidth transitive** command.

  However, if a 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 it to other peers.

f. Run **commit**

The configuration is committed.

----End

1.1.10.13.3 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** [*community-number* | *aa:nn*] &<1-29> [**internet** | **no-advertise** | **no-export** | **no-export-subconfed**] * [**whole-match**] command to check information about the routes carrying specified BGP community attributes.

----End

1.1.10.14 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**.

1.1.10.14.1 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 **1.1.12.10.2 (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

1.1.10.14.2 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

1.1.10.14.3 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

1.1.10.15 Configuring Prefix-based BGP ORF

Prefix-based BGP ORF enables a device to send its peer the local prefix-based import policy so that the peer can use the policy to filter routes before sending them to the local device.

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 Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

Step 4 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 5 Run the **peer** { *group-name* | *ipv4-address* } **capability-advertise orf** [**non-standard-compatible**] **ip-prefix** { **both** | **receive** | **send** } command to configure prefix-based ORF for a BGP peer or peer group.

Step 6 Run **commit**

The configuration is committed.

----End

Checking the Configurations

Run the following command to check the previous configuration.

- 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 routes received from a specified peer after the prefix-based ORF is configured.

1.1.10.16 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 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 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.

 Copyright © Huawei Technologies Co., Ltd.

Pre-configuration Tasks

Before setting parameters for a BGP peer connection, **configure basic BGP functions**.

1.1.10.16.1 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 set the hold time to a value greater than 20s. If the hold time is less than 20s, the peer session may be interrupted.

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 set the hold time to a value greater than 20s. If the hold time is less than 20s, the peer session may be interrupted.

d. Run **commit**

The configuration is committed.

----End

1.1.10.16.2 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

1.1.10.16.3 Disabling Fast Reset of EBGP Connections

Disabling rapid EBGP connection reset can prevent frequent reestablishment and deletion of EBGP sessions if route flapping occurs, which 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

1.1.10.16.4 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 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*

     A BGP ConnectRetry timer is configured for all 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*

     A BGP ConnectRetry timer is configured for a peer or peer group.

  d. Run **commit**

     The configuration is committed.

----End

1.1.10.16.5 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 BGP peers. If a network problem (congestion for example) occurs and slows down the speed at which the local

device advertises routes to a BGP peer in the update peer-group, the speed at which the local device advertises routes to other BGP peers in the update peer-group is affected. To address this problem, enable slow peer detection.

After slow peer detection is enabled, the local device calculates the difference between the time taken to send packets to each BGP peer and the shortest time taken to send packets to a BGP peer in the group. If the difference between the time taken to send packets to a BGP peer and the shortest time is greater than the threshold, the local device considers this BGP 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.

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*

Slow peer detection is enabled.

threshold *threshold-value* specifies a slow peer detection threshold. If the difference between the time taken to send packets to a BGP peer and the shortest time taken to send packets to another BGP peer is greater than the *threshold-value*, the local device considers the former BGP peer as a slow peer and removes it from the update peer-group.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.10.16.6 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

1.1.10.17 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.

NOTE

This step is not needed for dynamic IBGP peer groups.

Step 6 Run **peer** *group-name* **listen-net** *ipv4-address* { *mask* | *mask-length* }

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 BGP peers.

1.1.10.18 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 1-100 Configuring a device to send a default route to its peer

On the network shown in **Figure 1-100**, 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 a default route to the specified peer or a 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 based on the specified route policy.

If **conditional-route-match-all** { *ipv4-address1* { *mask1* | *mask-length1* } } &<1-4> is set, the BGP device sends the default route to the peer only when routes that match all the all specified conditions exist in the local routing table.

If **conditional-route-match-any** { *ipv4-address2* { *mask2* | *mask-length2* } } &<1-4> is set, the local device sends a default route to the peer when routes that match any of the specified conditions exist in the local 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

Checking the Configurations

After configuring a BGP device to send a default route to a peer, check whether the configuration is correct.

- Run the **display bgp routing-table** [*ipv4-address* [*mask* | *mask-length*]] command on a peer to check information about the received BGP default route.

1.1.10.19 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.

1.1.10.20 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 routing policy to modify the Local_Pref, AS_Path, Origin, or MED attribute of BGP routes to control traffic forwarding and 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.

 Copyright © Huawei Technologies Co., Ltd.

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* | *ipv6–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 igp-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.

    - 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 igp-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 igp-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 1-101**, 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 1-101 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.

Verifying the Configuration

After configuring BGP load balancing, verify the configuration.

- 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.

1.1.10.21 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.
  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 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, 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.

1.1.10.22 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 1-102**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-102 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 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**

The system view is displayed.

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 configuring the function, verify the configuration.

- 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.

1.1.10.23 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 1-103**, 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 1-103 Network without path MTU auto discovery

In **Figure 1-104**, 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 1-104 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

Checking the Configurations

After configuring path MTU auto discovery, run the following commands to check the previous configuration.

- Run the **display bgp peer** [*ipv4-address*] **verbose** command to check whether path MTU auto discovery has been successfully configured.

1.1.10.24 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 configuring BGP route recursion to the default route, run the **display current-configuration** command in the system view to check whether the function is configured.

1.1.10.25 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

NOTE

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

Checking the Configurations

Run the following commands to check the previous configuration.

- Run the **display bgp routing-table** *network* [*mask* | *mask-length*] command to check detailed information about a specified route in the BGP routing table.

1.1.10.26 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 1-105 is used as an example to show how the AIGP attribute is used during BGP route selection. In **Figure 1-105**, 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 1-105 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       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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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 1-19 shows the attribute comparison of the routes learned from DeviceA and DeviceB.

Table 1-19 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 Attribute	Route Learned from Device A	Route Learned from Device B	Comparison
Route type	Learned from a peer	Learned from a peer	The same.
AIGP	10	5	The different.

1.1.10.27 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 1-106**, 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 addressed to 10.1.1.0 from DeviceD, DeviceC removes the LSP label from the packet, searches for an outbound interface in the IP forwarding table according to the longest-match principle, and incorrectly sends the packet to DeviceB.

To solve the preceding problem, configure the **apply-label per-route pop-go** command 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 1-106 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

Checking the Configurations

After configuring the POPGO function, you can run the following command to check the configuration.

Run the **display mpls lsp protocol bgp** command. The command output shows detailed BGP LSP information. If **POPGO** is displayed in the **Label Operation** field, the POPGO function is successfully configured.

1.1.10.28 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 1-107**, 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 1-107 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**

     DeviceA is enabled to assign a label to each IPv4 unicast route.
  d. Run **quit**

     Return to the system view.
  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 1-108 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 1-108 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 DeviceA advertises routes to DeviceB.

h. Run **unicast-route label advertise**

DeviceA 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. Run **quit**

Return to the system view.

j. 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**]

DeviceB 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 configuring conversion from BGP IPv4 unicast routes to labeled routes, verify the configuration.

Run the **display bgp routing-table label** [**statistics**] command to check labeled route information in the BGP routing table.

1.1.10.29 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.

NOTE

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**] [**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 1-20 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 255.	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.

 Copyright © Huawei Technologies Co., Ltd.

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 BFD session check.

In scenarios where EBGP peers are indirectly connected, you can configure a TTL value for checking the BFD session with a specified BGP peer so that the traffic forwarding path can be quickly adjusted if BFD detects a fault on the link to the 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, 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 configuring BFD for BGP, you can run the following command to verify the configuration.

- Run the **display bgp bfd session** { [**vpnv4 vpn-instance** *vpn-instance-name*] **peer** *ipv4-address* | **all** } command to check information about the BFD session between BGP peers.

1.1.10.30 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* | *ipv6-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

1.1.10.31 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. During this period, services are interrupted. After the Router delivers the new optimal route to the FIB table, services are restored. Service interruption in this mode lasts a long time, which cannot 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.**
- **Configure BFD for 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 BGP IPv4 unicast address family view is displayed.

Step 4 Run **auto-frr**

BGP Auto FRR is enabled for unicast routes.

Step 5 (Optional) Run **route-select delay** *delay-value*

A delay for selecting a route to the intermediate device on the primary path is configured. After the primary path recovers, an appropriate delay ensures that

traffic switches back to the primary path after the intermediate device completes refreshing forwarding entries.

Step 6 Run **commit**

The configuration is committed.

----End

Checking the Configurations

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.

1.1.10.32 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 1-109**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-109 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 1-110**, 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 1-110 Networking diagram for a critical recursion change

- On the network shown in **Figure 1-111**, 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 1-111 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 Run **nexthop recursive-lookup delay** [*delay-time*]

The delay in responding to next hop recursion changes is configured.

The **nexthop recursive-lookup delay** command configures the device to delay responses to all recursion changes. The **nexthop recursive-lookup non-critical-event delay** command configures the device to delay responses only to non-critical BGP recursion changes. If both the commands are run, the **nexthop recursive-lookup non-critical-event delay** command takes precedence over the other command.

If both the commands are run, the delay time set using the **nexthop recursive-lookup non-critical-event delay** command must be greater than or equal to that set using the **nexthop recursive-lookup delay** command.

Step 4 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

After configuring the BGP next hop recursion changes delayed response, you can run the following command to check the previous configuration.

- Run the **display current-configuration configuration bgp | include nexthop recursive-lookup delay** command to view information about the delay in responding to a next hop recursion change.
- Run the **display current-configuration configuration bgp | include nexthop recursive-lookup non-critical-event delay** command to view information about the delay in responding to non-urgent next hop recursion changes.

1.1.10.33 Setting a Specified 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 BGP-IPv4 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 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* | *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 update-peer-group** command to check information about update peer-groups.

1.1.10.34 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 1-112**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-112 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 configuring a delay in releasing obtained labels in a BGP LSP FRR switchover scenario, verify the configuration.

Run the **display current-configuration** command to check the valid configuration.

1.1.10.35 Configuring the Route Server Function

This section describes how to configure the route server function. The function reduces network resource consumption onASBRs.

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.

1.1.10.36 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**.

1.1.10.36.1 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 the **ipv4-family flow** command to enter the BGP-Flow address family view, run the **ipv4-flow vpn-instance** command to enter the BGP-Flow VPN instance IPv4 address family view, run the **ipv4-flow vpnv4** command to enter the BGP-Flow VPNv4 address family view, run the **ipv6-family flow** command to enter the BGP-Flow IPv6 address family view, or run the **ipv6-flow vpn-instance** command to enter the BGP-Flow VPN instance IPv6 address family

view, and then run the **peer** { *ipv4-address* | *ipv6-address* } **graceful-restart static-timer** *restart-time* command to configure the maximum time for the local end to wait for GR recovery of the peer.

NOTE

ipv6-address takes effect only in the BGP-Flow IPv6 address family view and BGP-Flow VPN instance IPv6 address family view.

Step 7 Run **commit**

The configuration is committed.

----End

1.1.10.36.2 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

1.1.10.36.3 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

1.1.10.37 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.

 Copyright © Huawei Technologies Co., Ltd.

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 negotiates the GR capability only with the specified peers. If negotiation succeeds, BGP connections are established between the BGP 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** { *ipv4-address* | *group-name* } **capability-advertise graceful-restart**

GR is enabled for a specified peer or peer group, and the device is configured to advertise the GR capability to the peer or peer group.

NOTE

If the specified peer does not support GR, run the **peer** *ipv4-address* **local-graceful-restart enable** command instead to enable local GR for the peer.

If the specified peer group has peers that do not support GR, run both the **peer** *group-name* **capability-advertise graceful-restart** and **peer** *group-name* **local-graceful-restart enable** commands.

Step 4 (Optional) Run **peer** { *ipv4-address* | *group-name* } **graceful-restart timer restart** *time-value*

The maximum duration is set for the specified peer or peer group to wait for the BGP peer relationship to be reestablished with the local device. After the command is run, the local device will advertise the maximum duration to the specified peer or peer group.

NOTE

If the specified peer does not support GR, run the **peer** *ipv4-address* **local-graceful-restart timer restart** *restart-time* command instead to set the maximum duration for the local device to wait for the BGP peer relationship to be reestablished with the specified peer.

If the specified peer group has peers that do not support GR, run both the **peer** *group-name* **graceful-restart timer restart** *time-value* and **peer** *group-name* **local-graceful-restart timer restart** *restart-time* commands.

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 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 or peer group.

NOTE

If the specified peer does not support GR, run the **peer** *ipv4-address* **local-graceful-restart timer wait-for-rib** *wfrtime* command instead to set the maximum duration for the device to wait for the End-of-RIB flag from the specified peer.

If the specified peer group has peers that do not support GR, run both the **peer** *group-name* **graceful-restart timer wait-for-rib** *time-value* and **peer** *group-name* **local-graceful-restart timer wait-for-rib** *wfrtime* commands.

Step 7 Run **commit**

The configuration is committed.

----End

Verifying 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.

1.1.10.38 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 1-113 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 configuring BGP Best-external, verify the configuration.

- Run the **display bgp peer verbose** command to check whether the device has been enabled to advertise BGP Best-external routes to a specified peer or peer group.

1.1.10.39 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

In the BGP-VPNv4 address family view, the device can advertise Add-Path routes to IBGP and EBGP peers. However, in other address family views, Add-Path routes can be advertised only to IBGP peers.

BGP Add-Path is not supported in BGP confederation scenarios.

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 1-114**, 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 1-114 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**

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. (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* | *ipv6-address* | *group-name* } **capability-advertise add-path send**

The device is enabled to send Add-Path routes to a specified peer.

NOTE

Before running the **peer** *ipv6-address* **capability-advertise add-path { receive | send }** and **peer** *ipv6-address* **advertise add-path path-number** *number* { **route-policy** *route-policy-name* | **route-filter** *route-filter-name* } commands, you need to enter the IPv4 unicast address family view.

k. 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.

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* | *ipv6-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 completing the configurations, check the configurations on the device that advertises Add-Path routes.

- Run the **display bgp peer verbose** command to check information about BGP Add-Path.

1.1.10.40 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.**

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 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.

  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.

  Run **route-mode** { **ipv4-family unicast** | **ipv4-family labeled-unicast** | **ipv4-family vpnv4** } **adj-rib-out** { **pre-policy** | **post-policy** }

  The BMP device is configured to send statistics about RIB-out routes of BGP 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 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**]

BMP is configured to send statistics about Local-RIB routes of BGP peers in a specified address family to the monitoring server.

f. Run **quit**

Return to the BMP 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 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.

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 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.

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.

1.1.10.41 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 prevents frequent route flapping by using a penalty value to measure route stability. When a route flaps for the first time, a penalty value is assigned to the route. Later, each time the route flaps, the penalty value of the route increases by a specific value. The greater the penalty value, the less stable the route. 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, run the **ipv4-family vpnv4** command to enter the BGP-

VPNv4 address family view, or run the **ipv6-family vpnv6** 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.

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 BGP route dampening is configured, you can check whether the configuration is correct.

- 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 view BGP routes that flap within the specified time 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.

1.1.10.42 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, you have completed 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 configuring suppression on BGP peer flapping, verify the configuration.

- Run the **display bgp peer verbose** command to check the suppression status of the flapping BGP peer relationship and the remaining time to establish the BGP peer relationship.

1.1.10.43 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.

 Copyright © Huawei Technologies Co., Ltd.

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 vpnv4 routing-table** command to check BGP VPNv4 and BGP VPN route information.

1.1.10.44 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**]

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. 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. 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. 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-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** { *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 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 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 an RR 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 receive 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 receive from a peer.

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 configuring BGP-LS, 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.

1.1.10.45 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 1-115**, an end-to-end BGP LSP is established between PE1 and PE2. In addition, an end-to-end BGP-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. Load balancing technology is used to obtain higher bandwidth between nodes as a result of continuous network expansion. However, load imbalance becomes increasingly severe on the transit nodes. To improve load balancing performance, configure the entropy label capability for the BGP LSP on both PE1 and PE2.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-115 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 configuring the entropy label capability for a BGP LSP, verify the configuration.

- Run the **display bgp routing-table** command to check the entropy label in BGP routes.

1.1.10.46 Configuring BGP RPD

RPD provides a new method of distributing route-policies and allows the NCE 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 1-116**, 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 1-116 Typical networking of inbound traffic optimization

In this scenario, forwarders receive RPD policies delivered by the NCE and adjust route attributes and advertise routes based on the policies. The traffic optimization policy is configured on the NCE based on the traffic application. The following section describes relevant configurations on the forwarders. For details about how to configure the NCE, see the NCE manual.

Pre-configuration Tasks

Before configuring BGP RPD, complete the following tasks:

- **Configure basic BGP functions** on a device, and establish a connection between the device and NCE to ensure routing reachability between them.

Procedure

- Configure basic RPD functions.
  a. Run **system-view**

     The system view is displayed.
  b. Run **bgp** *as-number*

     The BGP view is displayed.
  c. Run **rpd-family**

     RPD is enabled, and the BGP RPD address family view is displayed.
  d. Run **peer** *ipv4–address* **enable**

     The device is enabled to exchange routing information with the specified peer.
  e. Run **quit**

     The BGP view is displayed.
  f. Run **ipv4-family unicast**

     The IPv4 unicast address family view is displayed.
  g. Run **peer** *ipv4-address* **rpd-policy export enable**

     The RPD export route-policy function is enabled in the IPv4 unicast address family view.
  h. Run **commit**

     The configuration is committed.
- (Optional) Configure GR to prevent the traffic interruption caused by a protocol restart.
  a. Run **system-view**

     The system view is displayed.
  b. Run **bgp** *as-number*

     The BGP view is displayed.
  c. Run **graceful-restart**

     GR is enabled in the BGP view.
  d. Run **rpd-family**

     The BGP RPD address family view is displayed.
  e. 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.
  f. 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 protocols to apply an updated RPD route-policy if the original policy changes.

a. Run **ipv4-family unicast**

The IPv4 unicast address family view is displayed.

b. 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.

1.1.10.47 Configuring IBGP Peers to Establish MPLS Local IFNET Tunnels

To carry BGP LSPs, you can configure IBGP peers to create an MPLS local IFNET tunnel.

Usage Scenario

To carry BGP LSPs, you can configure directly connected IBGP peers to establish an MPLS local IFNET tunnel. By default, IBGP peers cannot automatically establish an MPLS local IFNET tunnel. To enable the IBGP peers to establish an MPLS local IFNET tunnel, run the **undo peer mpls-local-ifnet disable** 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 **undo peer** { *peerGroupName* | *peerIpv4Addr* } **mpls-local-ifnet disable**

A specified IBGP peer is enabled to create an MPLS local IFNET tunnel.

NOTE

An MPLS local IFNET tunnel can be established between IBGP peers only in the BGP-IPv4 unicast address family or BGP unicast labeled address family. To allow an MPLS local IFNET tunnel to be established in the BGP-IPv4 unicast address family, run the **peer label-route-capability** command to enable the IBGP peers to send and receive labeled routes.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.10.48 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

1.1.10.49 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.

1.1.10.49.1 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.

1.1.10.49.2 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 NE40E-M2 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

Checking the Configurations

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.

1.1.10.49.3 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.

NOTE

Before performing this step, complete configuring basic keychain functions in **Pre-configuration Tasks** to create 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** *sndId* **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.

NOTE

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

1.1.10.49.4 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.

1.1.10.49.5 Configuring ROA

Resource Public Key Infrastructure (RPKI) ensures BGP security by validating the origin ASs of BGP 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 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 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**.

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:

  a. Run **system-view**

  The system view is displayed.

  b. Run **rpki**

  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 **rpki-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 **rpki-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**

Exit the RPKI session view.

j. Run **quit**

Exit the RPKI view.

k. Run **commit**

The configuration is committed.

NOTE

After RPKI session configurations are changed, run the **reset rpki 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 **rpki**

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**

Exit the RPKI origin-validation view.

f. Run **quit**

Exit the RPKI view.

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.

1.1.10.49.6 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**.

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 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**

Exit the RPKI region-validation-region view.

- 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.

Step 5 Run **bgp** *as-number*

The BGP view is displayed.

Step 6 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 7 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 8 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

Run the **display rpki session** *ipv4-address* **verbose** command to check RPKI session configurations.

1.1.10.49.7 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.

  NOTE

  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.

1.1.10.50 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.**

1.1.10.51 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 the VPN instance view.

Step 3 Run the **ipv4-family** command to configure a VPN instance IPv4 address family and enter its view.

Step 4 Run the **route-distinguisher** *route-distinguisher* command to set 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.

 Copyright © Huawei Technologies Co., Ltd.

Step 9 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

1.1.10.52 Maintaining BGP

Maintaining BGP involves resetting BGP connections and clearing BGP statistics.

1.1.10.52.1 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.

----End

 Copyright © Huawei Technologies Co., Ltd.

1.1.10.52.2 Clearing BGP Statistics

This section describes how to clear the statistics about flapping routes and dampened routes.

Context

NOTICE

BGP statistics cannot be restored after being cleared. Therefore, exercise caution when running the command.

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.
- 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

1.1.10.52.3 Configuring BGP to Record Peer Status Changes and Event Information

After you configure BGP to record peer status changes and event information, BGP logs every peer status change or event.

Context

If an error occurs on a BGP peer, BGP generates an error code and a subcode. If the error occurs on the local device, the local device disconnects the BGP peer and sends a BGP Notification message to the BGP peer. After the BGP peer receives the Notification message, it records the error code and subcode carried in the message and changes its state machine.

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. The system log files serve as a reference to locate network connectivity faults.

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

1.1.10.52.4 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

1.1.10.52.5 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 resolve this issue and enhance reliability, configure a special mode in which a device processes specified path attributes in received BGP Update messages. Special modes indicate those that are not defined in a standard protocol.

NOTE

This configuration may cause the specified path attributes to be dropped or the routes carrying the specified path attributes to be withdrawn. Therefore, exercise caution when you perform this configuration.

The **peer path-attribute-treat** command configuration takes effect immediately for the routes received after the command is run. However, for the routes received before the command is run, the configuration can take effect only after the **refresh bgp** command is run.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp path-attribute** { **originator-id** | **attr-set** } **accept-zero-value** or **bgp path-attribute** { **community** | **ext-community** | **ipv6-ext-community** | **large-community** | **attr-set** | **wide-community** } **accept-zero-length**

The method of handling incorrect path attributes is set.

BGP Update messages contain various path attributes. If the local device receives any Update message with an incorrect format, BGP session flapping may occur. To enhance reliability, you can perform this step to configure a special mode in which the device processes specified BGP path attributes. To configure the device to process path attributes with the value being 0, set **accept-zero-value**. To configure the device to process path attributes with the length being 0, set **accept-zero-length**.

NOTE

To configure the device to process incorrect BGP path attributes, perform this step. To configure the device to process incorrect BGP path attributes in a specific address family, enter the address family view and run the corresponding command.

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 the **peer** *peerIpv4Addr* **path-attribute-treat attribute-id** { *id* [**to** *id2*] } &<1-255> { **discard** | **withdraw** | **treat-as-unknown** } command to configure a processing mode for specified attributes. Alternatively, run the **peer** *peerIpv4Addr* **treat-with-error attribute-id** *id* **accept-zero-value** command to configure a processing mode for incorrect path attributes.

BGP Update messages contain various path attributes. If the local device receives any Update message with an incorrect format, BGP session flapping may occur. To enhance reliability, you can perform this step to configure a processing mode for specified BGP path attributes or incorrect path attributes.

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

The **treat-with-error** parameter specifies a processing mode for incorrect path attributes. The mode can be as follows:

- Accepting the path attributes with a value of 0.

Step 7 Run **commit**

The configuration is committed.

----End

1.1.10.52.6 Setting a Priority That Determines the Disconnection Order of a BGP Peer Relationship If Memory Overload Occurs

You can set a priority that determines the disconnection order of a BGP peer relationship upon memory overload. If the system memory usage exceeds the alarm threshold and the BGP memory usage is excessively high, such configurations allow BGP peer relationships to be disconnected in order of priority. This prevents BGP from exhausting the memory.

Usage Scenario

If BGP keeps accepting new routes in the case of a memory exception, the device memory may be exhausted. As a result, the involved process or board is reset. To prevent this problem, you can enable BGP to discard new routes upon a memory exception.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp** *as-number*

BGP is started (with the local AS number specified), and the BGP view is displayed.

Step 3 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 4 Run **ipv4-family unicast**

The BGP-IPv4 unicast address family view is displayed.

Step 5 Run **peer** *peerIpv4Addr* **memory-priority** *priority*

A priority is set, which determines the disconnection order of the specified BGP peer relationship if memory overload occurs.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

- Run the **display bgp peer verbose** command to check the priority that determines the disconnection order of the specified BGP peer relationship if memory overload occurs.

1.1.10.52.7 Enabling BGP to Discard New Routes If a Memory Exception Occurs

After BGP is enabled to discard new routes, BGP will discard the new routes received from peers in the case of a memory exception on the device.

Context

Usage Scenario

If BGP keeps accepting new routes when the device memory is abnormal, the device memory may be exhausted. As a result, the involved process or board is reset. To prevent this problem, you can enable BGP to discard new routes if a memory exception occurs.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp memory-usage exception-threshold discard new-route**

BGP is enabled to discard new routes if a memory exception occurs.

Step 3 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

- Run the **display bgp routing-table** command to check changes in the BGP routing table.

1.1.10.52.8 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.

1.1.10.52.9 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 following commands to check the previous 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.

NOTE

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.

1.1.10.52.10 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.

NOTE

If the function becomes abnormal or affects other services, you can run the **whitelist session-car rpki 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 following commands to check the previous 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.

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.

1.1.10.52.11 Configuring Micro-Segment CAR for BGP

You can configure micro-segment CAR for BGP to isolate bandwidth resources by interface and sub-interface for BGP messages used to establish peer relationships.

Context

When BGP messages used to establish peer relationships 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-segment CAR for BGP to isolate bandwidth resources by interface and sub-interface for the BGP messages. If the default parameters of micro-segment 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-segment 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-segment 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

1.1.10.53 BGP Route Selection Rules

1.1.10.53.1 Route Processing on the BGP Router

Understanding how the BGP Router processes routes helps you better control the BGP routing table.

Figure 1-117 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 before routes are selected, advertised, and delivered to the IP routing table.

Figure 1-117 Route processing on the BGP Router

Table 1-21 lists some key points in **Figure 1-117**.

Table 1-21 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.
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.

1.1.10.53.2 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.

Figure 1-118 shows how BGP selects the optimal route among multiple routes to the same destination on the NE40E.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-118 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 1-22** lists the abbreviated alias, route selection rules, and remarks of each matching item. **Table 1-22** 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 1-22 BGP route selection process

Abbreviated 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.

 Copyright © Huawei Technologies Co., Ltd.

Abbreviated Alias	Matching Item	Route Selection Rules	Remarks
A **NOTE** A is the initial of the character string (AS NIL).	Route generation mode	A > S > N > I > L, where: ● 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)	The route with the smallest AIGP value is preferred. The route with AIGP to the route without AIGP is preferred.	-
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	-

Abbreviated 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 igp-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, Cluster_List takes precedence over Originator_ID during BGP route selection. To enable Originator_ID to take precedence over Cluster_List during BGP 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 Origin AS.
- 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. If the **load-balancing med-ignore** command is run, BGP does not compare the MED attributes of routes when deciding the routes used for load balancing.
- All the routes are EBGP or IBGP routes. After the **maximum load-balancing eibgp** command is run, BGP does not use this rule in optimal VPN route selection.
- The metric values of the IGP routes to which BGP routes within an AS recurse are the same. After the **load-balancing igp-metric-ignore** command is run, the device does not compare IGP metric values when selecting routes for load balancing.
- All routes are blackhole or non-blackhole routes.

Even if the preceding conditions are met, load balancing cannot be implemented among labeled BGP routes and non-labeled BGP routes. Load balancing cannot be implemented between blackhole routes and non-blackhole routes.

VPN Route Selection Rules

On the NE40E, 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 NE40E *Feature Description > IP Routing > BGP*.

1.1.10.53.3 BGP Routing Table

This section describes how to check route attributes.

Table 1-23 lists all the common route attributes that affect route selection and the commands that are used to check them.

Table 1-23 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*]
Origin	**display bgp routing-table** [*network*]
MED	**display bgp routing-table** [*network*]

Route Attribute	Command Used to Check the Route Attribute
Peer type	**display bgp routing-table** *network*
IGP Metric	• **display bgp routing-table** *network* • **display ip routing-table** *ip-address* [*mask* | *mask-length*] [verbose], in which *ip-address* 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 1-24 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	Route status code, displayed in front of each route entry: ● *: 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. 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 packet), BGP adds a penalty value to the route carried in the packet. When the penalty value of a route exceeds the Suppress value, BGP suppresses the route by replacing the **>** sign of the route with the **d** or **h** sign. The route is ignored and its Update packets are not advertised to other BGP peers until the penalty value of the route decreases to the Reuse value. ● 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. 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. ● 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	Route Origin: ● 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.

 Copyright © Huawei Technologies Co., Ltd.

Item	Description
RPKI validation codes	Route origin AS validation code. ● 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 1-25 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: ● 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. ● 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.

 Copyright © Huawei Technologies Co., Ltd.

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
```

1.1.10.53.4 Route Attributes

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 modify the Next_Hop of the route.

Modifying the Next_Hop

In some scenarios, the Next_Hop needs to be modified. **Table 1-26** describes whether the Next_Hop needs to be modified in specific scenarios.

Table 1-26 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.	-

 Copyright © Huawei Technologies Co., Ltd.

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.	-
To prevent a device from modifying the Next_Hops of routes before advertising the routes to EBGP peers	**peer** { *group-name* | *ipv4-address* } **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.	-

 Copyright © Huawei Technologies Co., Ltd.

Objectives	Command	Usage Scenarios	Remarks
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.	-

 Copyright © Huawei Technologies Co., Ltd.

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** }	The Next_Hops of BGP routes can be modified using a route-policy in the following situations: ● 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 1-27**, the next hop is unreachable in the following situations:

Table 1-27 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: • 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 1-119 is used to show how to obtain a reachable next hop IP address. In **Figure 1-119**, 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 1-119 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        GigabitEthernet0/1/0
       10.1.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
      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.

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 1-28 lists two methods to modify the PreVal value.

Table 1-28 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 1-120**, both ISP1 and ISP2 advertise the routes 10.11.0.0/16 and 10.22.0.0/16 to AS 65001.

Figure 1-120 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: GigabitEthernet0/1/1
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: 0/1/0
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 1-29** shows the route attribute comparison of the routes learned from ISP1 and ISP2.

Table 1-29 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/1
 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: GigabitEthernet0/1/1
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: GigabitEthernet0/1/0
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.

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 1-30 lists two methods to modify the Local_Pref value.

Table 1-30 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 1-121 is used as an example to show how the Local_Pref value is used during BGP route selection. In **Figure 1-121**, both ISP1 and ISP2 advertise the routes 10.11.0.0/16 and 10.22.0.0/16 to AS 65001.

Figure 1-121 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                             0      100 10i
 * i                  10.1.2.1                100        0      200 10i
 *>   10.22.0.0/16      10.1.1.1                             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: GigabitEthernet0/1/1
Original nexthop: 10.1.1.1
Qos information : 0x0
AS-path 100 10, origin igp, 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: GigabitEthernet0/1/3
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igp, 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 1-31** shows the route attribute comparison of the routes learned from ISP1 and Device B.

Table 1-31 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: GigabitEthernet0/1/1
 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: GigabitEthernet0/1/0
 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 1-32** shows the route attribute comparison of the routes learned from Device A and Device B.

Table 1-32 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      100 10i
 *>   10.22.0.0/16       10.1.1.1                              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      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: GigabitEthernet0/1/4
 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: 00h22m23s
 Direct Out-interface: GigabitEthernet0/1/0
 Original nexthop: 10.1.2.1
 Qos information : 0x0
 AS-path 200 10, origin igp, 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 1-33** shows the route attribute comparison of the routes learned from Device A and ISP2.

Table 1-33 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	-	The route learned from Device A is optimal. **NOTE** If a route does not carry Local_Pref, the default value 100 takes effect.

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 1-122** as the networking):

Figure 1-122 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 10.1.1.1 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: GigabitEthernet0/1/3
 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: GigabitEthernet0/1/1
 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 1-34 shows the route attribute comparison of the routes learned from ISP1 and Device B.

Table 1-34 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: GigabitEthernet0/1/4
 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: GigabitEthernet0/1/0
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 1-123** as the networking):

Figure 1-123 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 10.1.2.1 route-policy rp2 export          //Apply the export policy rp2 to the routes from the peer
10.1.1.1 and modify the Local_Pref of the routes through rp2.
#
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      100 10i
* i                     10.1.2.1                  100        0      200 10i
*>   10.22.0.0/16       10.1.1.1                             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      i
*>   10.11.0.0/16       10.1.2.1                             0      200 10i
* i                     10.1.1.1                  100        0      100 10i
*>   10.22.0.0/16       10.1.2.1                             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                  120       0      100 10i
* i                     10.1.2.1                  100        0      200 10i
*>i  10.22.0.0/16       10.1.1.1                  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: GigabitEthernet0/1/0
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
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: GigabitEthernet0/1/1
Original nexthop: 10.1.2.1
Qos information : 0x0
AS-path 200 10, origin igp, 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.

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 1-124**, Device A and Device B are EBGP peers, and Device B, Device C, and Device D are IBGP peers.

Figure 1-124 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      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: GigabitEthernet0/1/4
 Original nexthop: 10.1.4.1
 Qos information : 0x0
 AS-path Nil, origin igp, 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: GigabitEthernet0/1/4
 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: GigabitEthernet0/1/4
 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.

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 1-125 is used as an example to show how the AIGP attribute is used during BGP route selection. In **Figure 1-125**, 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 1-125 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: GigabitEthernet0/1/1
 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/1
 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 1-35 shows the attribute comparison of the routes learned from Device A and Device B.

Table 1-35 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.

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_Confed_Sequence, and AS_Confed_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_Confed_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_Confed_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 1-36 describes the AS_Path-based route selection rules.

 Copyright © Huawei Technologies Co., Ltd.

Table 1-36 AS_Path-based route selection rules

Item	Description
AS_Set	BGP takes an AS_Set as one AS number even if the AS_Set contains multiple AS numbers. 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: ● 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	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. **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.
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).

 Copyright © Huawei Technologies Co., Ltd.

Item	Description
peer fake-as	After the command is run, BGP can use a fake AS number to set up a BGP peer relationship. 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.
peer substitute-as	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. **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.

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 1-126**, both ISP1 and ISP2 use 65001 as a private AS number.

Figure 1-126 Networking in which a private AS needs to be deleted

In **Figure 1-126**, 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 1-127**, AS 65005 imports three routes and advertises them to AS 65001 through two paths.

Figure 1-127 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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 1-37** shows the attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC.

Table 1-37 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 1-127**, 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 1-127**, 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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        GigabitEtherne0/2/0
       10.1.1.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/2/0
       10.1.3.0/30  Direct  0    0           D   10.1.3.1        GigabitEthernet0/1/0
       10.1.3.1/32  Direct  0    0           D   127.0.0.1       GigabitEthernet0/1/0
      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        GigabitEthernet0/2/0
                    EBGP    255  0           D   10.1.3.2        GigabitEthernet0/1/0
     172.16.2.0/24  EBGP    255  0           D   10.1.1.2        GigabitEthernet0/2/0
                    EBGP    255  0           D   10.1.3.2        GigabitEthernet0/1/0
     172.16.3.0/24  EBGP    255  0           D   10.1.1.2        GigabitEthernet0/2/0
                    EBGP    255  0           D   10.1.3.2        GigabitEthernet0/1/0
```

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.

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* } | **igp** | **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 1-128** as an example. In **Figure 1-128**, Device A and Device B are EBGP peers, and Device B, Device C, and Device D are IBGP peers.

Figure 1-128 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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 1-38** describes the attribute comparison of the routes learned from Device C and Device D.

Table 1-38 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: GigabitEthernet0/1/0
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: GigabitEthernet0/2/0
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 1-39** describes the attribute comparison of the routes learned from Device C and Device D.

Table 1-39 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.

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 1-40 lists three methods used to modify the MED value.

Table 1-40 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.

 Copyright © Huawei Technologies Co., Ltd.

Method	Usage Scenario
Configure an export policy in which the **apply cost-type med-inherit-aigp** command is run.	The Router sets the MED of each route to be advertised to a peer to the AIGP of the route. This method takes effect on both IBGP and EBGP peers. The policy applies only to BGP VPN peers.

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_Confed_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_Confed_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 1-129 is used as an example to describe how the MED attribute is used in BGP route selection. In **Figure 1-129**, ISP1 and ISP2 can access 1.1.1.9/32 from DeviceC or DeviceD.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-129 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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 1-41** describes the attribute comparison of the routes learned from DeviceC and DeviceD.

Table 1-41 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 1-130** shows the networking.

Figure 1-130 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 DeviceD:

NOTE

The following configurations are intended to ensure that DeviceA selects the route learned from DeviceC. However, DeviceA has already selected the route learned from DeviceC because the router ID of DeviceC is smaller than that of DeviceD. Therefore, the following configurations on DeviceD are optional.

```
#
bgp 65001
 #
 ipv4-family unicast
  undo synchronization
  peer 10.1.3.1 route-policy addmed200 export          //Apply 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 of addmed200 to allow
addmed200 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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 1-42 compares the attributes of the routes that DeviceB learns from DeviceC and DeviceD.

Table 1-42 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 1-130 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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, igp cost 13, internal, Router id 4.4.4.4
- Route A2: AS_Path { 65001 100 }, med 150, igp cost 11, internal, Router id 5.5.5.5
- Route B: AS_Path { 65002 300 }, med 0, igp 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.

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 1-131**, 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 1-131 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**.

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 igp-metric-ignore** command is configured,

BGP does not compare the IGP cost. In **Figure 1-132**, 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 1-132 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        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: GigabitEthernet0/2/0
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: GigabitEthernet0/1/0
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 1-43** describes the attribute comparison of the routes learned from Device A and Device B.

Table 1-43 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	The route learned from Device A is optimal. **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.

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 1-133**, 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 1-133 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
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 1-44** describes attribute comparison of the routes learned by DeviceA from DeviceB and DeviceC.

Table 1-44 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.

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 1-134**, 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 1-134 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: GigabitEthernet0/1/0
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, 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: GigabitEthernet0/2/0
Original nexthop: 10.1.2.2
Qos information : 0x0
AS-path Nil, origin igp, 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 1-45 describes the attribute comparison of the routes that DeviceA learns from DeviceB and DeviceC.

Table 1-45 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.

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 uniquely identifies a router in an AS, and the router ID can be configured as follows:

- Run the **router-id** { *ipv4-address* | **vpn-instance auto-select** } command. If no router ID is configured in the BGP view, BGP selects a router ID configured in the system view. For details on how a router ID configured in the system view is selected.
- Run the **router-id (BGP)** { *ipv4-address* | **auto-select** } command in the BGP VPN instance IPv4/IPv6 address family view. The **router-id (BGP VPN instance IPv4/IPv6 address family view)** command takes precedence over the **router-id (BGP)** command.

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. By default, Cluster_List takes precedence over Originator_ID during BGP route selection. To enable Originator_ID to take precedence over Cluster_List during BGP route selection, run the **bestroute router id-prior-clusterlist** command.

For more router ID-based route selection examples, see **Local_Pref**, **Origin**, and **MED**.

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

 Copyright © Huawei Technologies Co., Ltd.

relationships may be established between the two devices through the physical links.

In **Figure 1-135**, 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 1-135 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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.

1.1.10.53.5 (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 1-136**, 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 1-136 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.

1.1.10.53.6 Configuring BGP to Preferentially Select the Routes with Next Hops Recursing to SRv6 TE Policies

Usage Scenario

On the network shown in **Figure 1-137**, 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 1-137 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.

1.1.10.53.7 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.

 Copyright © Huawei Technologies Co., Ltd.

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.

1.1.10.54 Configuration Examples for BGP

This section provides BGP configuration examples.

1.1.10.54.1 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 1-138**, 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 1-138 Configuring basic BGP functions

NOTE

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

Device Name	Interface	IP Address
DeviceA	Loopback 0	1.1.1.1/32
	GE 0/1/0	172.16.0.1/16
	GE 0/2/0	192.168.0.1/24
DeviceB	Loopback 0	2.2.2.2/32
	GE 0/1/0	10.1.1.1/24
	GE 0/2/0	192.168.0.2/24
	GE 0/3/0	10.1.3.1/24
DeviceC	Loopback 0	3.3.3.3/32
	GE 0/2/0	10.1.2.1/24
	GE 0/3/0	10.1.3.2/24
DeviceD	Loopback 0	4.4.4.4/32
	GE 0/1/0	10.1.1.2/24

Device Name	Interface	IP Address
	GE 0/2/0	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
```

 Copyright © Huawei Technologies Co., Ltd.

```
[~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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.0.1 255.255.0.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.0.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.2 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 1-139** 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 1-139 Configuring BGP to interact with an IGP

NOTE

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

 Copyright © Huawei Technologies Co., Ltd.

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        GigabitEthernet0/1/0
       10.1.1.0/24  Direct 0    0           D   10.1.1.2        GigabitEthernet0/1/0
       10.1.1.1/32  Direct 0    0           D   10.1.1.1        GigabitEthernet0/1/0
       10.1.1.2/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
     10.1.1.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/1/0
       10.1.2.0/24  Direct 0    0           D   10.1.2.1        GigabitEthernet0/2/0
       10.1.2.1/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/2/0
     10.1.2.255/32  Direct 0    0           D   127.0.0.1       GigabitEthernet0/2/0
      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      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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  network 10.1.1.0 0.0.0.255
#
return
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
ospf 1
 area 0.0.0.0
  network 10.1.1.0 0.0.0.255
  network 10.1.2.0 0.0.0.255
#
return
```

1.1.10.54.3 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 1-140**, 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 1-140 Configuring the MED attribute to control route selection

NOTE

Interfaces 1 through 2 in this example are GE 0/1/0 and GE 0/2/0, 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: GigabitEthernet0/1/0
 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: GigabitEthernet0/2/0
 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: GigabitEthernet0/2/0
 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: GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ip address 172.16.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.4 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 1-141**, 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 1-141 Configuring BGP route filtering

NOTE

Interfaces 1 through 2 in this example are GE 0/1/0 and GE 0/2/0, 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      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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
#
```

```
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.5 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 1-142**, 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 1-142 Configuring BGP RRs

NOTE

Interfaces 1 through 5 in this example represent GE 0/1/0, GE 0/2/0, GE 0/3/0, GE 0/1/1, and GE 0/1/2, respectively.

Table 1-46 IP addresses of the interfaces

Device	Interface	IP Address	Device	Interface	IP Address
Device A	GE 0/3/0	172.16.1.1/24	Device C	GE 0/1/1	10.1.8.1/24
	GE 0/1/0	10.1.1.2/24		GE 0/1/2	10.1.9.1/24
	GE 0/2/0	10.1.3.2/24	Device D	GE 0/1/0	10.1.4.2/24
Device B	GE 0/1/0	10.1.1.1/24		GE 0/2/0	10.1.6.1/24
	GE 0/2/0	10.1.4.1/24	Device E	GE 0/2/0	10.1.6.2/24
	GE 0/3/0	10.1.5.1/24		GE 0/3/0	10.1.5.2/24
	GE 0/1/1	10.1.2.1/24	Device F	GE 0/1/0	10.1.7.2/24
Device C	GE 0/1/0	10.1.2.2/24	Device G	GE 0/1/0	10.1.8.2/24
	GE 0/2/0	10.1.3.1/24	Device H	GE 0/2/0	10.1.9.2/24
	GE 0/3/0	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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 10.1.5.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 10.1.7.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.8.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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.

1.1.10.54.6 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 1-143**, 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 1-143**, 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 1-143 Configuring the confederation

NOTE

Interfaces 1 through 5 in this example represent GE0/1/0, GE0/2/0, GE0/3/0, GE0/1/1, and GE0/1/2, 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
```

 Copyright © Huawei Technologies Co., Ltd.

```
[~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: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/0
 Original nexthop: 10.1.3.1
 Qos information : 0x0
 AS-path 100, origin igp, 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.216.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 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 GigabitEthernet0/1/0
 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
```

NOTE

The configuration file of DeviceC is similar to that of DeviceB.

- DeviceD configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.3.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

NOTE

The configuration file of DeviceE is similar to that of DeviceD.

- DeviceF configuration file

```
#
sysname DeviceF
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.216.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.7 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 1-144**, 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 1-144 Configuring the BGP community attribute

NOTE

Interfaces 1 through 3 in this example are GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 andDevice 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: GigabitEthernet0/2/0
 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: GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.5.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.2.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/3/0
 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
```

1.1.10.54.8 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 1-145**, 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 1-145 Networking diagram for configuring prefix-based BGP ORF

NOTE

Interfaces 1 through 3 in this example are GE 0/1/0, GE 0/2/0, GE 0/3/0, 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 1-145**. 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.

```
[~A] 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 GigabitEthernet0/2/0
 ip address 10.2.1.2 255.255.255.252
#
interface GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/2/0
 ip address 10.3.1.2 255.255.255.252
#
interface GigabitEthernet0/1/0
 ip address 10.1.1.2 255.255.255.252
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/1
 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
```

1.1.10.54.9 Example for Configuring BGP Route Dampening

Configuring BGP route dampening can improve network stability.

Networking Requirements

In **Figure 1-146**, 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. It is required that BGP route dampening be configured to suppress unstable routes and improve network stability.

Figure 1-146 Configuring BGP route dampening

NOTE

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

Precautions

When configuring BGP route dampening, note the following rules:

- 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 Device A and Device C, between Device B and Device C, and between Device D and Device C.
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 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 the routes that flap.

```
[*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 Device C.

```
[~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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.10.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.10.2.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.10.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.10.2.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 10.10.3.2 255.255.255.0
```

 Copyright © Huawei Technologies Co., Ltd.

```
#
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
#
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
#
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
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.10 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 1-147**, 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 1-147 Configuring BGP default route advertisement

NOTE

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

Device Name	Interface	IP Address
Device A	GE 0/1/0	10.20.1.1/24
	Loopback 0	1.1.1.1/32
Device B	GE 0/1/0	10.20.1.2/24
	GE 0/2/0	10.0.1.1/24
	GE 0/3/0	10.0.3.2/24
	Loopback 0	2.2.2.2/32
Device C	GE 0/1/0	10.20.2.2/24
	GE 0/2/0	10.0.1.2/24
	GE 0/3/0	10.0.2.1/24
	Loopback 0	3.3.3.3/32
Device D	GE 0/1/0	10.20.3.2/24
	GE 0/2/0	10.0.3.1/24
	GE 0/3/0	10.0.2.2/24
	Loopback 0	4.4.4.4/32
Device E	GE 0/1/0	10.20.2.1/24
	GE 0/2/0	10.1.1.1/24
	Loopback 0	5.5.5.5/32

Device Name	Interface	IP Address
Device F	GE 0/1/0	10.20.3.1/24
	GE 0/2/0	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

 Copyright © Huawei Technologies Co., Ltd.

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 10.0.1.0 0.0.0.255
[*DeviceB-ospf-1-area-0.0.0.0] network 10.0.3.0 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 Device C.

```
[~DeviceC] ospf 1
[*DeviceC-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 10.0.1.0 0.0.0.255
[*DeviceC-ospf-1-area-0.0.0.0] network 10.0.2.0 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 Device D.

```
[~DeviceD] ospf 1
[*DeviceD-ospf-1] area 0
[*DeviceD-ospf-1-area-0.0.0.0] network 10.0.2.0 0.0.0.255
[*DeviceD-ospf-1-area-0.0.0.0] network 10.0.3.0 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 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 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 as-path-filter 10 permit ^(400_)+$
[*DeviceD] ip as-path-filter 10 permit ^(400_)+_[0-9]+$
[*DeviceD] ip ip-prefix default permit 0.0.0.0 0
[*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 GigabitEthernet0/1/0
 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
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.20.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.0.1.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.20.2.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.0.1.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 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
```

```
 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 GigabitEthernet0/1/0
 undo shutdown
  ip address 10.20.3.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.0.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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
 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
#
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
#
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]+$
#
return
```

- Device E configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.20.2.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.20.3.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
 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
```

1.1.10.54.11 Example for Configuring BGP Load Balancing

Configuring load balancing can fully utilize network resources and reduce network congestion.

Networking Requirements

In **Figure 1-148**, 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 1-148 Configuring BGP load balancing

NOTE

Interfaces 1 through 2 in this example are GE 0/1/0 and GE 0/2/0, 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: GigabitEthernet0/1/0
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: GigabitEthernet0/2/0
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: GigabitEthernet0/1/0
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: GigabitEthernet0/2/0
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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

 Copyright © Huawei Technologies Co., Ltd.

```
 undo synchronization
 peer 10.1.2.1 enable
 peer 10.1.4.1 enable
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 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
```

1.1.10.54.12 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 1-149**, OSPF is used as the IGP in AS 100. An IBGP peer relationship is established between loopback0s of DeviceA and DeviceB, and between loopback0s 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 GE0/1/0 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 GE0/1/2 as the outbound interface, causing traffic loss.

Figure 1-149 Networking diagram for configuring BGP next hop recursion based on a routing policy

NOTE

Interfaces 1 through 4 in this example are GE 0/1/0, GE 0/1/1, GE 0/1/2, 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-0.0.0.0] network 1.1.1.1 0.0.0.0
[*DeviceA-ospf-1-area-0.0.0.0] network 10.1.0.0 0.0.255.255
[*DeviceA-ospf-1-area-0.0.0.0] commit
[~DeviceA-ospf-1-area-0.0.0.0] quit
[~DeviceA-ospf-1] quit
```

Configure Device B.

```
[~DeviceB] ospf 1
[*DeviceB-ospf-1] area 0
[*DeviceC-ospf-1-area-0.0.0.0] network 2.2.2.2 0.0.0.0
[*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 3.3.3.3 0.0.0.0
[*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] commit
[~DeviceC-ospf-1-area-0.0.0.0] quit
[~DeviceC-ospf-1] quit
```

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 permit 0.0.0.0 32
[*DeviceA] route-policy np-by-rp permit node 0
[*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: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/1
 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 0/1/0 of Device B to simulate a fault on Device B.

```
[~DeviceB] interface GigabitEthernet 0/1/0
[~DeviceB-GigabitEthernet0/1/0] shutdown
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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: GigabitEthernet0/1/1
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
```

```
#
interface GigabitEthernet0/1/2
 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 0.0.0.0
  network 1.1.1.1 0.0.0.0
  network 10.1.0.0 0.0.255.255
#
route-policy np-by-rp permit node 10
 if-match ip-prefix np-by-rp-ip
#
ip ip-prefix np-by-rp-ip index 10 permit 0.0.0.0 32
#
return
```

- DeviceB configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 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 0.0.0.0
  network 2.2.2.2 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 GigabitEthernet0/1/1
 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
```

1.1.10.54.13 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 1-150**, 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 1-150 Network diagram of filtering routes to be advertised and accepted

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.2.1 255.255.255.0
#
bgp 200
 router-id 2.2.2.2
 peer 192.168.1.2 as-number 200
 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

1.1.10.54.14 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 1-151**, 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 1-151 Configuring BFD for BGP

NOTE

Interfaces 1 through 3 in this example are GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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
[*DeviceA-bgp] peer 10.20.2.2 as-number 200
[*DeviceA-bgp] peer 10.20.2.2 ebgp-max-hop
[*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
[*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
[*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          GigibitEthernet0/1/0
 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 0/2/0 of DeviceB to simulate a fault on the primary link.

```
[~DeviceB] interface gigabitethernet 0/2/0
[*DeviceB-GigabitEthernet0/2/0] shutdown
[*DeviceB-GigabitEthernet0/2/0] 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.20.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.20.1.2 255.255.255.0
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
```

```
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.15 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 1-152**, 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 1-152 Configuring BGP Auto FRR

NOTE

Interfaces 1 through 3 in this example represent GE 0/1/0, GE 0/2/0, 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: GigabitEthernet0/1/0
     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: GigabitEthernet0/1/0
     TunnelID: 0x0                Flags:  D
    BkNextHop: 10.2.1.2       BkInterface: GigabitEthernet0/2/0
      BkLabel: 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.2.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.3.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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
```

1.1.10.54.16 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 1-153**, 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 1-153 Networking for configuring BGP ADD-PATH

NOTE

Interfaces 1 through 6 in this example represent GE 0/3/0, GE 0/3/2, GE 0/3/3, GE 0/1/1, GE 0/1/2, and GE 0/1/3, 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 1-47**

Table 1-47 Configurations of each device

Device	Router ID	Interface	IP Address	AS Number
Device A	1.1.1.1	GigabitEthernet 0/3/0	172.16.3.1/24	AS 65008

Device	Router ID	Interface	IP Address	AS Number
		GigabitEthernet 0/1/1	172.16.2.1/24	
		GigabitEthernet 0/1/3	172.16.1.1/24	
Device B	2.2.2.2	GigabitEthernet 0/3/0	172.16.3.2/24	AS 65008
		GigabitEthernet 0/3/2	172.16.7.1/24	
		GigabitEthernet 0/1/2	172.16.5.2/24	
Device C	3.3.3.3	GigabitEthernet 0/3/0	172.16.6.1/24	AS 65008
		GigabitEthernet 0/3/3	172.16.4.2/24	
		GigabitEthernet 0/1/3	172.16.1.2/24	
Device D	4.4.4.4	GigabitEthernet 0/3/0	172.16.6.2/24	AS 65009
		GigabitEthernet 0/3/2	172.16.7.2/24	
		LoopBack0	1.1.1.1/32	
RR	5.5.5.5	GigabitEthernet 0/3/3	172.16.4.1/24	AS 65008
		GigabitEthernet 0/1/1	172.16.2.2/24	
		GigabitEthernet 0/1/2	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: GigabitEthernet0/1/1
 Original nexthop: 172.16.7.2
 Qos information : 0x0
 AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, inte
rnal, 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: GigabitEthernet0/1/1
 Original nexthop: 172.16.7.2
 Qos information : 0x0
 AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, inte
rnal, 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: GigabitEthernet0/1/1
 Original nexthop: 172.16.6.2
 Qos information : 0x0
 AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, inte
rnal, 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: GigabitEthernet0/1/2
 Original nexthop: 172.16.7.2
 Qos information : 0x0
 AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, inte
rnal, 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: GigabitEthernet0/3/3
 Original nexthop: 172.16.6.2
 Qos information : 0x0
 AS-path 65009, origin incomplete, MED 0, localpref 100, pref-val 0, valid, inte
rnal, 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 GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.3.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 172.16.2.1 255.255.255.0
#
interface GigabitEthernet0/1/3
 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 GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.3.2 255.255.255.0
```

```
#
interface GigabitEthernet0/3/2
 undo shutdown
 ip address 172.16.7.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 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 GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.6.1 255.255.255.0
#
interface GigabitEthernet0/3/3
 undo shutdown
 ip address 172.16.4.2 255.255.255.0
#
interface GigabitEthernet0/1/3
 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 GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.6.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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 GigabitEthernet0/3/3
 undo shutdown
 ip address 172.16.4.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 undo shutdown
 ip address 172.16.2.2 255.255.255.0
#
interface GigabitEthernet0/1/2
 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
```

1.1.10.54.17 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 1-154**, 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 1-154 Networking diagram of configuring BGP keychain authentication

NOTE

Interface 1 in this example is GE 0/1/0.

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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

1.1.10.54.18 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 1-155**, 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 1-155 Configuring BGP-LS

NOTE

Interfaces 1 through 4 in this example represent GE0/1/0, GE0/2/0, GE0/3/0, and GE0/1/8, 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
- 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 0/2/0
[*DeviceA-GigabitEthernet0/2/0] isis enable 1
[*DeviceA-GigabitEthernet0/2/0] commit
[~DeviceA-GigabitEthernet0/2/0] 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 0/1/8
[*DeviceB-GigabitEthernet0/1/8] isis enable 1
[*DeviceA-GigabitEthernet0/1/8] commit
[~DeviceB-GigabitEthernet0/1/8] quit
```

Configure Device C.

```
[~DeviceC] isis 1
[*DeviceC-isis-1] network-entity 10.0000.0000.0003.00
[*DeviceC-isis-1] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis enable 1
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] interface gigabitethernet 0/3/0
[*DeviceC-GigabitEthernet0/3/0] isis enable 1
[*DeviceC-GigabitEthernet0/3/0] quit
[*DeviceC] interface gigabitethernet 0/1/8
[*DeviceC-GigabitEthernet0/1/8] isis enable 1
[*DeviceC-GigabitEthernet0/1/8] commit
[~DeviceC-GigabitEthernet0/1/8] 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 0/3/0
[*DeviceD-GigabitEthernet0/3/0] isis enable 1
[*DeviceD-GigabitEthernet0/3/0] 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    GE0/1/0           Direct          D/-/L/-
10.1.1.0/24        10         NULL    GE0/2/0           Direct          D/-/L/-
10.1.2.0/24        10         NULL    GE0/1/8           Direct          D/-/L/-
192.168.0.0/24     10         NULL    GE0/3/0           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    GE0/1/0           Direct          D/-/L/-
10.1.1.0/24        10         NULL    GE0/2/0           Direct          D/-/L/-
10.1.2.0/24        10         NULL    GE0/1/8           Direct          D/-/L/-
172.16.1.1/32      10         NULL    GE0/3/0           192.168.0.2     A/-/-/-
192.168.0.0/24     10         NULL    GE0/3/0           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
[*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-id0000.0000.0003.02]][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 :  ?
 *>    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                           LocPrf    :
      MED      : 0                              PrefVal   : 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                           LocPrf    :
      MED      : 0                              PrefVal   : 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                           LocPrf    :
```

```
      MED      : 0                                PrefVal   : 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                            LocPrf    :
      MED      : 0                                PrefVal   : 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                            LocPrf    :
      MED      : 0                                PrefVal   : 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                            LocPrf    :
      MED      : 0                                PrefVal   : 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                            LocPrf    :
      MED      : 0                                PrefVal   : 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                            LocPrf    :
      MED      : 0                                PrefVal   : 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                            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-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix1.1.1.0/24]]
      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-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix10.1.1.0/24]]
      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-id0000.0000.0003.00]][PREFIX[ospf-route-type0][prefix10.1.2.0/24]]
      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-id0000.0000.0003.00]][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 GigabitEthernet0/2/0
 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 GigabitEthernet0/1/8
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 1.1.1.1 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.0.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/1/8
 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 GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 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
```

1.1.10.54.19 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 usage 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 1-156**, BGP runs on all Routers. Router A and Router B reside in AS 100, Router C in AS 200, and NCE 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, NCE 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 1-156 Networking for configuring BGP RPD

NOTE

Interface 1 and interface 2 in this example stand for GE 0/1/0 and GE 0/1/1, 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 NCE.
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. For details about NCE's configurations, such as the BGP, BGP RPD address family, and traffic optimization policy configurations, see the NCE manual.

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
[*DeviceB-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: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/1
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 NCE 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.2.1.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.3.1.1 255.255.255.0
#
interface GigabitEthernet0/1/1
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.2.1.2 255.255.255.0
#
interface GigabitEthernet0/1/1
 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
```

1.1.10.54.20 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 1-157**, 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 1-157 Network diagram of configuring dynamic BGP peer groups

NOTE

In this example, interface 1, interface 2, interface 3, and interface 4 represent GE0/1/1, GE0/1/2, GE0/1/3, and GE0/1/4, 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.1.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet0/1/3
 undo shutdown
 ip address 10.2.1.1 255.255.255.0
interface GigabitEthernet0/1/4
 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 GigabitEthernet0/1/1
 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 GigabitEthernet0/1/1
 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 GigabitEthernet0/1/1
 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 GigabitEthernet0/1/1
 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
```

1.1.10.54.21 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 1-158**, the public network BGP-IPv4 unicast address family is enabled in base BGP instances on DeviceA and DeviceB. An EBGP peer relationship is established in the public network BGP-IPv4 unicast address family to transmit public network routes. In addition, the VPN address family is enabled in the BGP multi-instance (an extended BGP instance) on DeviceB and in the base BGP instance on DeviceC. An EBGP-VPN peer relationship is established in the VPN address family to transmit VPN routes. In this way, routes can be managed and maintained on a per BGP instance basis.

Figure 1-158 BGP multi-instance networking

NOTE

In this example, interfaces 1 and 2 refer to GE 0/1/0 and GE 0/2/0, 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 BGP 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 gigabitethernet0/1/0
[*DeviceA-GigabitEthernet0/1/0] ip address 10.1.1.1 24
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 gigabitethernet0/1/0
[*DeviceB-GigabitEthernet0/1/0] ip address 10.1.1.2 24
[*DeviceB-GigabitEthernet0/1/0] commit
[~DeviceB-GigabitEthernet0/1/0] 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, run the **ping** command to check the connectivity between DeviceA and DeviceB. Then, run the **display bgp peer** command to check the EBGP peer relationship. The command output shows that the public network EBGP peer relationship between DeviceA and DeviceB is **Established**.

The following example uses the ping result and command output on DeviceB.

```
<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 BGP 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 aa
[~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] interface gigabitethernet0/2/0
[*DeviceB-GigabitEthernet0/2/0] ip binding vpn-instance vpn1
[*DeviceB-GigabitEthernet0/2/0] ip address 10.1.2.2 24
[*DeviceB-GigabitEthernet0/2/0] 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] interface gigabitethernet0/2/0
[*DeviceC-GigabitEthernet0/2/0] ip binding vpn-instance vpn1
[*DeviceC-GigabitEthernet0/2/0] ip address 10.1.2.3 24
[*DeviceC-GigabitEthernet0/2/0] commit
```

After the configuration is complete, run the **ping** command to check the connectivity between DeviceB and DeviceC. Then, run the **display bgp instance aa vpnv4 all peer** command to check the EBGP-VPN peer relationship. The command output shows that the EBGP-VPN peer relationship between DeviceB and DeviceC is **Established**.

The following example uses the ping result and command output on DeviceB.

```
<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 vpn1, Router ID 10.1.1.2:
 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 each of DeviceA and DeviceC.

Configure DeviceA.

```
[~DeviceA] ip route-static 192.168.1.1 32 NULL 0
[*DeviceA] bgp 100
[*DeviceA-bgp] import-route static
[*DeviceA-bgp] commit
```

Configure DeviceC.

```
[~DeviceC] ip route-static vpn-instacne vpn1 192.168.3.3 32 NULL 0
[*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 a public network IPv4 route is displayed in the BGP routing table 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       ?
```

A public network IPv4 route is displayed in the base BGP instance's routing table, and an IPv4 VPN route is displayed in the BGP multi-instance routing table 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:

Total Number of Routes: 1
      Network          NextHop                MED       LocPrf    PrefVal Path/Ogn

*>    192.168.3.3/32   10.1.2.3                0                0     100?
```

Only a VPN route is displayed in the routing table of the BGP VPN instance 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:

Total Number of Routes: 1
      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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/2/0
 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
```

1.1.10.54.22 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 1-159**, OSPF runs between DeviceA and DeviceB, between DeviceB and DeviceC, and between DeviceD and DeviceE. IS-IS runs between DeviceC and DeviceD. Basic MPLS functions and MPLS LDP are configured on DeviceA through DeviceE so that LDP LSPs are established between loopback interfaces of devices in each IGP area. In this case, 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, interface1 and interface2 represent GE0/1/0 and GE0/2/0, respectively.

Figure 1-159 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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ip address 10.1.1.1 24
[*DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable
[*DeviceC-GigabitEthernet0/1/00] quit
[*DeviceC] interface loopBack 1
[*DeviceC--LoopBack1] isis enable
[*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 0/1/0
[*DeviceA-GigabitEthernet0/1/0] mpls
[*DeviceA-GigabitEthernet0/1/0] mpls ldp
[*DeviceA-GigabitEthernet0/1/0] commit
[~DeviceA-GigabitEthernet0/1/0] 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 GigabiteEthernet0/1/0. 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      :  GigabiteEthernet0/1/0
  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      :  GigabiteEthernet0/2/0
  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 **GigabiteEthernet0/1/0**. 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      :  GigabiteEthernet3/0/1
  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      :  GigabiteEthernet0/1/0
  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 32 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 32 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
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         18050/20050   -/-
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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.2.2 255.255.255.0
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
 isis enable 1
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.0
 mpls
 mpls ldp
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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 GigabitEthernet0/2/0
 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
```

1.1.10.54.23 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 1-160**, 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 GE0/1/0 and GE0/2/0, respectively.

Figure 1-160 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 0/1/0
[~DeviceA-GigabitEthernet0/1/0] ipv6 enable
[*DeviceA-GigabitEthernet0/1/0] ipv6 address 2001:db8:44::1 96
[*DeviceA-GigabitEthernet0/1/0] 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 0/1/0
[~DeviceB-GigabitEthernet0/1/0] ipv6 enable
[*DeviceB-GigabitEthernet0/1/0] ipv6 address 2001:db8:11::1 96
[*DeviceB-GigabitEthernet0/1/0] quit
[~DeviceB] interface gigabitethernet 0/2/0
[~DeviceB-GigabitEthernet0/2/0] ipv6 enable
```

```
[*DeviceB-GigabitEthernet0/2/0] ipv6 address 2001:db8:44::2 96
[*DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[~DeviceC-GigabitEthernet0/1/0] ipv6 enable
[*DeviceC-GigabitEthernet0/1/0] ipv6 address 2001:db8:22::1 96
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface gigabitethernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ipv6 enable
[*DeviceC-GigabitEthernet0/2/0] ipv6 address 2001:db8:11::2 96
[*DeviceC-GigabitEthernet0/2/0] 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 0/1/0
[~DeviceD-GigabitEthernet0/1/0] ipv6 enable
[*DeviceD-GigabitEthernet0/1/0] ipv6 address 2001:db8:33::1 96
[*DeviceD-GigabitEthernet0/1/0] quit
[*DeviceD] interface gigabitethernet 0/2/0
[*DeviceD-GigabitEthernet0/2/0] ipv6 enable
[*DeviceD-GigabitEthernet0/2/0] ipv6 address 2001:db8:22::2 96
[*DeviceD-GigabitEthernet0/2/0] 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 0/2/0
[~DeviceE-GigabitEthernet0/2/0] ipv6 enable
[*DeviceE-GigabitEthernet0/2/0] ipv6 address 2001:db8:33::2 96
[*DeviceE-GigabitEthernet0/2/0] 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 0/1/0
```

```
[*DeviceA-GigabitEthernet0/1/0] isis ipv6 enable 1
[*DeviceA-GigabitEthernet0/1/0] 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 0/2/0
[*DeviceB-GigabitEthernet0/2/0] isis ipv6 enable 1
[*DeviceB-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis ipv6 enable 100
[*DeviceC-GigabitEthernet0/1/0] 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 0/2/0
[*DeviceD-GigabitEthernet0/2/0] isis ipv6 enable 100
[*DeviceD-GigabitEthernet0/2/0] 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* GE0/1/0            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              Direct                     0        D/-/L/-
28
2001:DB8:4::4/1 GE0/1/0            FE80::865B:12FF:FE75:E73D  10       A/-/-/-
28
2001:DB8:11::/9 GE0/2/0            Direct                     10       D/-/L/-
6
2001:DB8:22::/9 GE0/1/0            Direct                     10       D/-/L/-
6
    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 0/2/0
[*DeviceC-GigabitEthernet0/2/0] ip binding vpn-instance vrftest
[*DeviceC-GigabitEthernet0/2/0] 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 0/1/0
[*DeviceD-GigabitEthernet0/1/0] ip binding vpn-instance vrftest
[*DeviceD-GigabitEthernet0/1/0] 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 vpnv6
[*DeviceC-bgp-af-vpnv6] 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                    Binding SID         : -
TunnelType             : SRv6-TE Policy              DelayTimerRemain     : -
Policy State           : Up                    State Change Time    : 2022-02-11 17:09:59
Admin State            : Up                      Traffic Statistics   : Disable
```

```
Backup Hot-Standby      : Disable                    BFD                 : Disable
Interface Index         : -                          Interface Name      : -
Interface State         : -                          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                : 21                         Policy Name         : policy1
 Template ID            : 0                          Path Verification   : Disable
 DelayTimerRemain       : -                          Network Slice ID    : -
 Segment-List Count     : 1
  Segment-List          : s1
   Segment-List ID      : 21                         XcIndex             : 21
   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:DB8:444::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-vrftest-af-ipv6] commit
[~DeviceD-vpn-instance-vrftest-af-ipv6] quit
[~DeviceD-vpn-instance-vrftest] 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 vrftest** 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 vrftest
Routing Table : vrftest
         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    : GigabitEthernet0/1/0              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    : GigabitEthernet0/1/0              Flags        : D

Destination  : 2001:DB8:11::2                    PrefixLength : 128
NextHop      : ::1                             Preference   : 0
Cost         : 0                             Protocol     : Direct
RelayNextHop : ::                                TunnelID     : 0x0
Interface    : GigabitEthernet0/1/0              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 vrftest 2001:DB8:5::5 verbose
Route Flags: R - relay, D - download to fib, T - to vpn-instance, B - black hole route
------------------------------------------------------------------------------
Routing Table : vrftest
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 00:00:36 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.

NOTE

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              : 1.11.1.1
 Remote Router Id             : 4.44.4.4
 Local Interface Address      : 2001:DB8:2::2
 Remote Interface Address     : 2001:DB8:5::5
 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                : GigabitEthernet0/1/0

 Peer Adj                     : 2001:DB8:11::2
 Local ASN                    : 200
 Remote ASN                   : 300
 Local Router Id              : 1.11.1.1
 Remote Router Id             : 4.44.4.4
 Interface Identifier         : 24
 Local Interface Address      : 2001:DB8:11::1
 Remote Interface Address     : 2001:DB8:11::2
 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                : GigabitEthernet0/1/0
[~DeviceE]display bgp egress-engineering

 Peer Node                    : 2001:DB8:2::2
 Peer Adj Num                 : 1
 Local ASN                    : 300
 Remote ASN                   : 200
 Local Router Id              : 4.44.4.4
 Remote Router Id             : 1.11.1.1
 Local Interface Address      : 2001:DB8:5::5
 Remote Interface Address     : 2001:DB8:2::2
 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                : GigabitEthernet0/2/2

 Peer Adj                     : 2001:DB8:33::1
 Local ASN                    : 300
 Remote ASN                   : 200
 Local Router Id              : 4.44.4.4
 Remote Router Id             : 1.11.1.1
 Interface Identifier         : 17
 Local Interface Address      : 2001:DB8:33::2
 Remote Interface Address     : 2001:DB8:33::1
 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                : GigabitEthernet0/2/2
```

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                     Binding SID         : -
TunnelType           : SRv6-TE Policy         DelayTimerRemain    : -
Policy State         : Up                     State Change Time   : 2022-02-11 18:15:17
Admin State          : Up                     Traffic Statistics  : Disable
Backup Hot-Standby   : Disable                BFD                 : Disable
Interface Index      : -                      Interface Name      : -
```

```
Interface State         : -                          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 GigabitEthernet0/3/0
 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
 ipv6 import-route bgp level-1
 #
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:11::1/96
#
interface GigabitEthernet0/2/0
 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 1.11.1.1
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:22::1/96
 isis ipv6 enable 100
#
interface GigabitEthernet0/2/0
 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 vpnv6
  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 GigabitEthernet0/1/0
 undo shutdown
 ip binding vpn-instance vrftest
 ipv6 enable
 ipv6 address 2001:DB8:33::1/96
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:DB8:22::2/96
 isis ipv6 enable 1
#
interface LoopBack1
 ipv6 enable
 ipv6 address 2001:DB8:4::4/128
 isis ipv6 enable 1
#
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 vpnv6
  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 GigabitEthernet0/2/0
 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 100
 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 vpnv6
  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
```

1.1.10.54.24 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 1-161**, 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 GE0/1/1, GE0/1/2, and GE0/1/3, respectively.

Figure 1-161 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 0/1/1
[~DeviceA-GigabitEthernet0/1/1] ip address 10.1.1.1 24
[*DeviceA-GigabitEthernet0/1/1] 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 10.1.1.2 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
[*DeviceB-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: GigabitEthernet0/1/3
 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: GigabitEthernet0/1/2
 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: GigabitEthernet0/1/2
 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: GigabitEthernet0/1/3
 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 GigabitEthernet0/1/1
 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 undo shutdown
 ip address 10.1.1.2 255.255.255.0
```

```
#
interface GigabitEthernet0/1/3
 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.3.1 255.255.255.0
 ospf enable 1 area 0.0.0.0
#
interface GigabitEthernet0/1/2
 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 GigabitEthernet0/1/1
 undo shutdown
 ip address 10.1.4.1 255.255.255.0
#
interface GigabitEthernet0/1/2
 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
```

1.1.11 BGP4+ Configuration

BGP4+ transmits routing information between Autonomous Systems (ASs) and is applicable to large-scale and complex IPv6 networks.

1.1.11.1 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 used for packet forwarding.
- Multiprotocol Unreachable NLRI (MP_UNREACH_NLRI): carries the set of unreachable destinations.

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 BGP4, BGP4+ is applicable to IPv6 networks without changing the messaging and routing mechanisms of BGP4.

Purpose

BGP transmits route information between ASs. It, however, is not required in all scenarios.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-162 BGP networking

BGP is required in the following scenarios:

- On the network shown in **Figure 1-162**, 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. Routers, 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 NE40E-M2 series Feature Description - VPN*.
- Users need to transmit multicast routes and construct a multicast topology. For details, see the *HUAWEI NE40E-M2 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.

1.1.11.2 Configuration Precautions for BGP4+

Feature Requirements

None

1.1.11.3 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.

1.1.11.3.1 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

NOTICE

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 set.

NOTE

If the router ID of the Router is the IP address of a physical interface, the change in the IP address will cause route flapping on the network. To enhance network stability, configuring the address of a loopback interface as the router ID is recommended.

If the device does not have an IPv4 interface, it needs to be configured with a router ID.

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

1.1.11.3.2 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:

  a. Run **system-view**

     The system view is displayed.

  b. Run **bgp** *as-number*

     The BGP view is displayed.

c. 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 loopback interface on a reachable 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 IPv6 address used to set up a BGP4+ peer relationship is a loopback interface address, follow the procedure described in **Configuring the Local Interface for the BGP4+ Connection**.

d. (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.

e. (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.

f. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

g. 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.

h. 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. 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 loopback interface on a reachable 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 IP address used to set up a BGP4+ peer relationship is a loopback interface address, or a local-link address, follow the procedure described in **Configuring the Local Interface for the BGP4+ Connection**.

d. 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.

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 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.

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. (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.

j. Run **commit**

The configuration is committed.

----End

1.1.11.3.3 (Optional) Configuring the Local Interface for a BGP4+ Connection

When loopback interfaces are used to establish a BGP4+ connection between two peers, the local interface used for the BGP4+ connection needs to be configured on the devices.

Context

To improve the reliability and stability of a BGP4+ connection, the local interfaces used for the BGP4+ connection should be configured as loopback interfaces. When redundant links are available on the network, the BGP4+ connection will not be torn down if an interface or a link fails.

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* | *group-name* } **connect-interface** *interface-type interface-number* [*ipv6-source-address*]

The source interface and source address used to set up a TCP connection are specified.

BGP4+ uses the physical interface that is directly connected to the peer as the local interface of a TCP connection.

NOTE

When multiple peer relationships between two Routers are established through various links, running the **peer connect-interface** command to specify the local interface for a BGP4+ connection is recommended.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.11.3.4 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** *ipv6-address* **verbose** command to check detailed information about BGP4+ peers.
- Run the **display bgp ipv6 peer** *ipv6-address* **log-info** command to check the log information of BGP4+ peers.
- Run the **display bgp ipv6 routing-table** command to check the information about BGP4+ routes.

----End

1.1.11.4 Configuring a BGP4+ Peer Group

Configuring a BGP4+ peer group simplifies the management of routing policies and improves route advertisement efficiency.

Usage Scenario

A large-scale BGP4+ network has a large number of peers. It is difficult to configure and maintain them. To address this problem, peer groups can be configured to simplify the management of routing policies and improve the efficiency of route advertisement. Peer groups are classified as IBGP peer groups and EBGP peer groups depending upon whether the peers reside in the same AS or not, respectively. EBGP peer groups are further divided into pure EBGP peer groups and mixed EBGP peer groups. If all the peers in an EBGP peer group reside in the same external AS, the group is a pure EBGP peer group; otherwise, the group is a mixed EBGP peer group.

 Copyright © Huawei Technologies Co., Ltd.

Pre-configuration Tasks

Before configuring a BGP4+ peer group, **configure basic BGP4+ functions**.

Procedure

- Configure an IBGP peer group.

  If BGP4+ has multiple IBGP peers, adding them to an IBGP peer group can simplify routing policy management. When creating an IBGP peer group, you do not need to specify the AS number.

  a. Run **system-view**

     The system view is displayed.

  b. Run **bgp** *as-number*

     The BGP view is displayed.

  c. Run **group** *group-name* [**internal**]

     A peer group is created.

  d. Run **ipv6-family unicast**

     The IPv6 unicast address family view is displayed.

  e. Run **peer** *group-name* **enable**

     The peer group is enabled.

  f. Run **peer** { *ipv4-address* | *ipv6-address* } **group** *group-name*

     An IPv6 peer is added to the peer group.

     NOTE

     After an IBGP peer is added to a peer group, the system automatically creates an IPv6 peer in the BGP view and enables the IPv6 peer in the IPv6 address family view.

  g. Run **commit**

     The configuration is committed.

- Configure a pure EBGP peer group.

  If BGP4+ has multiple EBGP peers that belong to the same AS, adding them to a pure EBGP peer group can simplify routing policy management. In a pure EBGP peer group, all the peers must reside in the same AS.

  a. Run **system-view**

     The system view is displayed.

  b. Run **bgp** *as-number*

     The BGP view is displayed.

  c. Run **group** *group-name* **external**

     A peer group is created.

  d. Run **peer** *group-name* **as-number** *as-number*

     The AS number is specified for the peer group.

e. Run **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

f. Run **peer** *group-name* **enable**

The peer group is enabled.

g. Run **peer** *ipv6-address* **group** *group-name*

An IPv6 peer is added to the peer group.

After an EBGP peer is added to a peer group, the system automatically creates an EBGP peer in the BGP view and enables the EBGP peer in the IPv6 address family view.

An AS number needs to be specified for the pure EBGP peer group.

If the peer group already has peers in it, no AS number needs to be specified for the peer group.

h. Run **commit**

The configuration is committed.

- Configure a mixed EBGP peer group.

  If BGP4+ has multiple EBGP peers that belong to different ASs, adding them to a mixed EBGP peer group can simplify the routing policy management. When creating a mixed EBGP peer group, specify an AS number for each peer.

  a. Run **system-view**

  The system view is displayed.

  b. Run **bgp** *as-number*

  The BGP view is displayed.

  c. Run **group** *group-name* **external**

  A peer group is created.

  d. Run **peer** *ipv6-address* **as-number** *as-number*

  The AS number is specified for an IPv6 peer.

  e. Run **ipv6-family unicast**

  The IPv6 unicast address family view is displayed.

  f. Run **peer** *group-name* **enable**

  The peer group is enabled.

  g. Run **peer** *ipv6-address* **group** *group-name*

  The created IPv6 peer is added to the peer group.

  After the EBGP peer is added to the peer group, the system automatically enables the EBGP peer in the IPv6 address family view.

  When creating a mixed EBGP peer group, you need to create peers and specify their AS numbers separately. No AS number needs to be specified for the mixed EBGP peer group.

  h. Run **commit**

The configuration is committed.

----End

Checking the Configuration

After the configuration is complete, check whether the configuration has taken effect.

- Run the **display bgp ipv6 group** [*group-name*] command to check information about peer groups.

1.1.11.5 Controlling Route Import

BGP4+ can import the routes discovered by other protocols into the BGP4+ routing table. When importing IGP routes, BGP can filter routes by routing protocol.

Usage Scenario

BGP4+ cannot discover routes by itself. Instead, it imports routes discovered by other protocols, such as OSPFv3 or static routes into 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.

Pre-configuration Tasks

Before controlling route import, **configure basic BGP4+ functions**.

1.1.11.5.1 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.

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.

BGP can import routes in either Import or Network mode:

- Import mode: BGP4+ imports routes by protocol type, such as RIP, OSPF, IS-IS, static routes, and direct routes.
- Network mode: BGP4+ imports a route with the specified prefix and mask to the BGP4+ routing table. Network mode is more precise than 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.

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.

e. 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*]

A local IPv6 route is imported.

If no mask or mask length is specified, the address is considered as a classful address.

The local routes to be advertised must be in the local IP routing table. You can use routing policies or XPL route filters to flexibly control the routes to be advertised.

NOTE

- The destination address and mask specified in the **network** command must be consistent with 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

1.1.11.5.2 Configuring BGP4+ to Import Default Routes

This section describes how to configure BGP4+ to import default routes from the local 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 **default-route imported**

BGP is configured to import default routes.

To import default routes, run both the **default-route imported** command and the **import-route** command. If only the **import-route** command is used, default routes cannot be imported. In addition, the **default-route imported** command is used to import only the default routes that exist in the local routing table.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.11.5.3 Verifying the Configuration

After controlling route import is configured, verify information about filtered and imported routes.

Prerequisites

Configurations have been performed to control route import.

Procedure

- Run the **display bgp ipv6 routing-table** [*ipv6-address prefix-length*] command to check BGP4+ routes.
- 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 the specified AS_Path filter.

----End

1.1.11.6 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**.

1.1.11.6.1 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* | *mask-length* } command.
- To advertise only summary routes, run the **aggregate** *ipv6-address* { *mask* | *mask-length* } **detail-suppressed** command.
- To advertise some of the specific routes, run the **aggregate** *ipv6-address* { *mask* | *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* | *mask-length* } **as-set** command.
- To configure the attributes of summary routes, run the **aggregate** *ipv6-address* { *mask* | *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* | *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

1.1.11.6.2 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>]

Default routes are sent to a remote 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 based on the specified route policy.

If **conditional-route-match-all** { *ipv6-address1 ipv6-mask-length1* } &<1-4> is set, the BGP device sends the default route to the peer only when routes that match all the all specified conditions exist in the local IPv6 routing table.

If **conditional-route-match-any** { *ipv6-address2 ipv6-mask-length2* } &<1-4> is set, the local device sends a default route to the peer when routes that match any of the specified conditions exist in the local IPv6 routing table.

NOTE

After the **peer default-route-advertise** command is run, BGP4+ sends a default route with the local address as the next hop to the specified peer, regardless of whether there are default routes in the routing table.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.11.6.3 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

1.1.11.6.4 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.

NOTE

- 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

1.1.11.6.5 Setting the Interval at Which Update Packets Are Sent

When routes change, the Router sends Update packets to notify its peers. If a route changes frequently, set an interval at which Update packets are sent to prevent the Router from sending Update packets for every change.

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 for sending Update messages is set.

Step 5 Run **commit**

The configuration is committed.

----End

1.1.11.6.6 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** | **verbose** } command to check the log information about BGP4+ peers.

----End

1.1.11.7 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**.

1.1.11.7.1 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 learned from peers in other ASs
- IBGP routes learned from peers in the same AS
- Routes generated locally 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

1.1.11.7.2 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

1.1.11.7.3 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

1.1.11.7.4 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.
  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 **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

1.1.11.7.5 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 vpnv6** [**unicast**]

  The BGP-VPNv6 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 1-163**, a BGP LSP is established between PE1 and PE2. VPNv6 routes are exchanged between PE1 and RR1, between RR1 and RR2, and between RR2 and PE2 through BGP VPNv6 peer relationships.

  Figure 1-163 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 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 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

1.1.11.7.6 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, or run the **ipv6-family unicast** command to enter the BGP-IPv6 unicast 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

1.1.11.7.7 Configuring AIGP Attributes for Routes

The Accumulated Interior Gateway Protocol Metric (AIGP) attribute allows devices in an AIGP administrative domain to use the optimal routes to forward data.

Context

An AIGP administrative domain is a set of autonomous systems (ASs) in a common administrative domain.

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 the **ipv6-family unicast** command to enter the BGP-IPv6 unicast address family view or run the **ipv6-family vpnv6** command to enter the BGP-VPNv6 address family view.

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. If a BGP peer with the AIGP capability joins a BGP peer group that does not have the AIGP capability, the BGP peer still retains the AIGP capability. 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 VPN routes to participate in route selection using the AIGP attribute of the BGP LSP through which they are transmitted.

1. Run **system-view**

   The system view is displayed.

2. Run **bgp** *as-number*

   The BGP view is displayed.

3. Run **ipv6-family vpn-instance** *vpn-instance-name*

   The BGP-VPN instance IPv6 address family view is displayed.

4. Run **bestroute nexthop-resolved aigp**

   VPN routes are allowed to participate in route selection using the AIGP attribute of the BGP LSP through which they are transmitted.

5. Run **commit**

   The configuration is committed.

Step 3 (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

1.1.11.7.8 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* &<1-13>] [**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

1.1.11.8 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**.

1.1.11.8.1 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. 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 list filter

  Before configuring a conditional BGP4+ route advertisement policy, you need to create an IPv6 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 1-48 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. ● Examples of matched strings: 65, 651, 6501, and 65001 ● Examples of unmatched strings: 165, 1650, 6650, and 60065

Special Character	Function	Example
$	Indicates the end of a matched string.	65$ matches strings ending with 65. ● 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 only. **NOTE** ^$ matches an empty string (empty AS_Path) and is usually used to match routes in the local AS.
_	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 ($).	● ^65001_ matches the AS_Paths that begin with 65001 followed by a sign. Specifically, ^65001_ matches AS_Paths with 65001 as the leftmost AS number (the number of the last AS through which a route passes) and the routes sent by peers in AS 65001. ● _65001_ matches the strings (AS_Paths) that contain 65001, which is used to match the routes that pass through AS 65001. ● _65001$ 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), which is used to match the routes that originate in AS 65001.

 Copyright © Huawei Technologies Co., Ltd.

Special Character	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: ● \(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. ● \(.*_65003_.*\) matches the AS_Confed_Sequence that contains AS number 65003 and the routes that pass through AS 65003 in a confederation. ● _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\). Similarly, backslashes (\) can be used to remove the special meanings of the left bracket ([) and right bracket (]) used

Special Character	Function	Example
		in an AS_Confed_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.	65* matches the AS_Paths that begin with 6 and contain zero or multiple 5s. ● 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.	65+ matches the AS_Paths that begin with 6 and contain one or multiple 5s. ● 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.	65? matches the AS_Paths that begin with 6 and contain zero or one 5. ● Examples of matched strings: 6 and 65 ● Examples of unmatched strings: 655, 6559, and 65529
()	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.

 Copyright © Huawei Technologies Co., Ltd.

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. **NOTE** 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 the *HUAWEI NE40E-M2 seriesUniversal Service Router Configuration Guide - Basic Configurations*.

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** | **no-advertise** | **no-export** | **no-export-subconfed**] &<1-20> or **ip community-filter** *cfIndex* [**index** *index-val*] *matchMode* [*cmntyStr* | *cmntyNum* | **internet** | **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** } { *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 **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** { *as-number:nn* | *4as-number:nn* | *ipv4-address:nn* } } &<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 prefix list.

Before configuring a conditional BGP4+ route advertisement policy, you need to create an IPv6 prefix list.

a. Run **system-view**

The system view is displayed.

b. Run **filter-list ipv6-prefix** *name*

An IPv6 prefix list is created, and the IPv6 prefix list view is displayed.

c. Run **prefix** *address maskLen*

An IPv6 address and mask are configured for the IPv6 prefix list.

d. Run **commit**

The configuration is committed.

----End

1.1.11.8.2 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** clauses

  The clauses define the matching rules of a route-policy. The matching objects are route attributes.

- **apply** clauses

  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.

NOTE

This section describes only the BGP4+ route-policy. For detailed information about route-policy configurations, see "Routing Policy Configuration."

Procedure

- Create a 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 view of the route-policy is displayed.
  c. Run **commit**

     The configuration is committed.
- Configure **if-match** clauses.
  a. 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 in Step 3 can be configured in random order. 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.
  b. Run **commit**

     The configuration is committed.
- Configure the **apply** clause.
  a. Run **system-view**

The system view is displayed.

b. Run **route-policy** *route-policy-name matchMode* **node** *node*

The route-policy view is displayed.

c. 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 NE40E-M2 series Universal Service 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.

d. Run **commit**

The configuration is committed.

----End

1.1.11.8.3 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 routes to be advertised are filtered.

BGP4+ filters the routes imported using the **import-route** command, and only the routes that meet the matching rules are added to the local BGP4+ routing table and advertised to BGP4+ peers.

If *protocol* is specified, only the routes of the specified protocol are matched. If *protocol* is not specified, all BGP routes to be advertised are matched, 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.

- To filter routes based on a basic ACL, perform the following steps:

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

1.1.11.8.4 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:

     - To filter routes based on a basic ACL, perform the following steps:
       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 received 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 received 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 router does not retry setting up a connection. An alarm is generated and a log is recorded. In this case, run the **display bgp peer** [**verbose**]

command, and you can find that the status of the peer is Idle. To restore the BGP connection, run the **reset bgp** command.

- **idle-timeout**: The peer relationship is interrupted. The router retries setting up a connection after the timer expires. An alarm is generated and a log is recorded. In this case, run the **display bgp peer** [**verbose**] command, and 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 router 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

1.1.11.8.5 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.

Procedure

- Enable the route-refresh capability.

  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** { *ipv4-address* | *ipv6-address* | *group-name* } **capability-advertise route-refresh**

  The route-refresh capability is enabled.

  If the route-refresh capability has been enabled on all BGP4+ Routers and a BGP4+ routing policy is changed, the local Router sends route-refresh messages to its peers. After receiving the messages, the peers resend their routing information to the local Router. Based on the received routing information, the local Router can refresh its BGP4+ routing table dynamically and apply the new routing policy, without tearing down any BGP4+ connections.

  d. Run **commit**

  The configuration is committed.

- Configure a BGP4+ device to store all the routing updates received from its peers.

  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.

- Softly reset a BGP4+ connection.

  Perform the following steps on a BGP4+ device:

  a. Run **refresh bgp ipv6** { **all** | *ipv4-address* | *ipv6-address* | **group** *group-name* | **external** | **internal** } { **export** | **import** }

     A BGP4+ connection is softly reset.

     To perform a soft reset on a BGP4+ connection, perform this step in the user view.

----End

1.1.11.8.6 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** *ipv6-address* { **advertised-routes** | **received-routes** } [**statistics**] command to view the routes advertised to or received from a specified BGP4+ peer.

----End

1.1.11.9 Configuring BGP4+ XPL

BGP4+ extended routing-policy language (XPL) route-filters can be used to flexibly control BGP4+ acceptance and advertisement.

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.**

1.1.11.9.1 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

1.1.11.9.2 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

1.1.11.10 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.

1.1.11.11 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 1-164**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-164 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.

1.1.11.12 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 igp-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.
   - Run the **load-balancing as-path-ignore** command to prevent the device from comparing AS_Path attributes when selecting routes for load balancing.
   - 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.
   - Run the **load-balancing igp-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.

- Run the **load-balancing as-path-ignore** command to prevent the device from comparing AS_Path attributes when selecting routes for load balancing.
- 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.
- Run the **load-balancing igp-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 1-165**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-165 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 **ipv6-family vpn-instance** *vpn-instance-name*

The BGP-VPN instance IPv6 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 BGP4+ routes carrying the Link Bandwidth 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 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.

1.1.11.13 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 1-166**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-166 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.

1.1.11.14 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**.

1.1.11.14.1 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 set the hold time to a value greater than 20s. If the hold time is less than 20s, the peer session may be interrupted.

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

1.1.11.14.2 Enabling Fast Reset of EBGP Connections

With fast reset of EBGP connections, BGP4+ rapidly detects EBGP link faults and resets EBGP connections on related 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 reset of EBGP connections is enabled.

After this function is enabled, BGP4+ rapidly detects EBGP link faults and resets EBGP connections on related interfaces.

If an interface on which a BGP4+ connection is established alternates between Up and Down, run the **undo ebgp-interface-sensitive** command to prevent repeated re-establishment and deletion of the BGP4+ session in the case of route flapping. This saves network bandwidth.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.11.14.3 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 specific peer or 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

1.1.11.14.4 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** [**verbose**] command to check information about BGP4+ peers.

----End

1.1.11.15 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 x (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 1-167 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 1-167 Networking with hierarchical RRs

Pre-configuration Tasks

Before configuring BGP4+ RRs, **configure basic BGP4+ functions**.

1.1.11.15.1 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

1.1.11.15.2 (Optional) Disabling Route Reflection Between Clients Through the RR

If the clients of an RR are fully meshed, you can disable route reflection among them through the RR to reduce bandwidth consumption.

Context

On some networks, if the clients of an RR have established IBGP peer relationships with each other, they can directly exchange routing information. Route reflection between the clients through the RR is unnecessary. 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 **ipv6-family unicast**

The IPv6 unicast address family view is displayed.

Step 4 Run **undo reflect between-clients**

Route reflection among 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 among 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

1.1.11.15.3 (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

1.1.11.15.4 (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

1.1.11.15.5 (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 path attributes of BGP4+ routes using an export policy.

After the **reflect change-path-attribute** command is run on an RR, route attributes can be modified using an export policy. Configurations that take effect include:

- Run the **apply as-path** command to modify the AS_Path attribute of BGP4+ routes.
- Run the **apply comm-filter delete** command to delete the community attribute of BGP4+ routes.
- Run the **apply community** command to modify the community attribute of BGP4+ routes.
- Run the **apply large-community** command to modify the Large-community attribute of BGP4+ routes.
- Run the **apply cost** command to modify the MED value of BGP4+ routes.
- Run the **apply ipv6 next-hop** command to modify the next-hop IP address of BGP4+ routes.
- Run the **apply local-preference** command to modify the Local_Pref attribute of BGP4+ routes.
- Run the **apply origin** command to modify the origin attribute of BGP4+ routes.
- Run the **apply extcommunity** command to modify the VPN target extended community attribute of BGP4+ 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

1.1.11.15.6 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** [**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

1.1.11.16 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:

  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*

     The confederation ID is set.

d. 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.

1.1.11.17 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:

- **1.1.11.3 Configuring Basic BGP4+ Functions**

1.1.11.17.1 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 **1.1.12.10.2 (Optional) Configuring an if-match Clause**.
  d. Configure community or extended community attributes for BGP4+ routes.
     - 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.
  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. 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

1.1.11.17.2 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 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 either of the following commands as required to advertise community attributes to a specified peer.

- To enable the device to advertise a standard community attribute to a specified peer, run the **peer** *ipv6-address* **advertise-community** command.
- To enable the device to advertise an extended community attribute to a specified peer, run the **peer** *ipv6-address* **advertise-ext-community** command.

After the **peer advertise-ext-community** command is run, BGP4+ advertises the 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. 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 either of the following commands as required to advertise community attributes to a specified peer.

- To enable the device to advertise a standard community attribute to a specified peer, run the **peer** *ipv6-address* **advertise-community** command.
- To enable the device to advertise an extended community attribute to a specified peer, run the **peer** *ipv6-address* **advertise-ext-community** command.

  After the **peer advertise-ext-community** command is run, BGP4+ 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 from the received routing information.

  To enable the device to advertise the Link Bandwidth extended community attribute to a specified EBGP peer, run the **peer advertise ebgp link-bandwidth** command.

  To enable the device to convert the Link Bandwidth extended community attribute (optional non-transitive) carried in BGP4+ routes to an optional transitive attribute before advertising the routes to other peers, run the **peer advertise link-bandwidth transitive** command.

f. Run **commit**

The configuration is committed.

----End

1.1.11.17.3 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

1.1.11.18 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 messages to a peer to detect the status of the peer. 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**] [**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* } *

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.

1.1.11.19 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. During this period, services are interrupted. After the Router delivers the new optimal route to the FIB table, services are restored. Service interruption in this mode lasts a long time, which cannot 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**.
- **Configure BFD for BGP4+**

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 **auto-frr**

BGP4+ Auto FRR is enabled for unicast routes.

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.

1.1.11.20 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.

1.1.11.21 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. However, establishing full-mesh connections among devices that function as ASBRs 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 in IBGP scenarios and allows devices to advertise routes to their clients (ASBR devices) without changing route attributes, such as AS_Path, Nexthop, and MED. With the route server function, EBGP full-mesh connections are not required among the ASBR devices, which reduces network resource consumption.

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.

1.1.11.22 Configuring the BGP4+ GR Helper

A BGP4+ GR helper helps its neighbor complete BGP4+ GR.

Usage Scenario

BGP4+ restart causes re-establishment of peer relationships and 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 the GR capability to establish a GR-capable BGP4+ session.

Pre-configuration Tasks

Before configuring BGP4+ GR helper, **configure basic BGP4+ functions**.

1.1.11.22.1 Enabling BGP4+ GR

Enabling or disabling BGP4+ GR may delete and re-establish all BGP4+ 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

1.1.11.22.2 Configuring BGP4+ GR Session Parameters

Changing the BGP4+ restart period re-establishes BGP4+ 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*

The period during which the restarting speaker and receiving speaker wait for End-Of-RIB messages is set.

NOTE

You can adjust BGP4+ GR session parameter values as needed. However, retaining default BGP4+ GR session parameter values is recommended.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.11.22.3 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

1.1.11.23 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 the RR that needs to advertise Add-Path routes:
  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**

     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 the device (DeviceA in this example) 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.

1.1.11.24 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.

 Copyright © Huawei Technologies Co., Ltd.

- **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 vpnv6** } **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 vpnv6** } **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 vpnv6** } **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.

1.1.11.25 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 peer. After this command is run, the device will advertise the GR capability to the specified peer.

NOTE

If a specified peer does not support GR, run the **peer** *ipv6-address* **local-graceful-restart enable** command instead to enable local GR for the specified peer.

Step 4 (Optional) Run **peer** *ipv6-address* **graceful-restart timer restart** *time-value*

The maximum duration is set for the specified peer to reestablish its BGP4+ peer relationship with the device. After the command is run, the device will advertise the maximum duration to the specified peer.

NOTE

If a specified peer does not support GR, 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

 Copyright © Huawei Technologies Co., Ltd.

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.

NOTE

If a specified peer does not support GR, run the **peer** *ipv6-address* **local-graceful-restart timer wait-for-rib** *wfrtime* 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.

1.1.11.26 Configuring BGP4+ Route Dampening

Configuring BGP4+ route dampening suppresses instable BGP4+ routes.

Usage Scenario

BGP4+ route dampening is designed to suppress unstable routes and improve 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.

A primary cause of route instability is route flapping. A route is considered to be flapping when it repeatedly appears and then disappears in the routing table. BGP4+ is applied to complex networks where routes change frequently. Frequent route flapping consumes lots of bandwidth and CPU resources and even seriously affects network operations. To prevent the impact of frequent route flapping, BGP4+ uses route dampening to suppress unstable routes.

With specified-requirement route dampening, you can use a route-policy to differentiate routes so that BGP4+ can apply 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 vpnv6** 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 vpnv6** 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 vpnv6 routing-table dampening parameter** command to check configured BGP VPNv6 route dampening parameters.
- Run the **display bgp vpnv6 routing-table dampened** command to check dampened BGP VPNv6 routes.

1.1.11.27 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 when the BGP4+ 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 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, you have completed the following task:

- **1.1.11.3 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** *peerIpv6Addr* **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.

1.1.11.28 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 vpnv6 routing-table** command to check BGP VPNv6 and BGP VPN route information.

1.1.11.29 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**]

     IS-IS topology advertisement is enabled.

     viii. (Optional) Run **bgp-ls identifier** *identifier-value*

     A BGP-LS identifier is configured for IS-IS.

     ix. 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. 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. 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.

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** *ipv6-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.

In most cases, the BGP-LS routing table contains a large number of BGP-LS routes. If a lot of BGP-LS routes are received from peers, excessive system resources will be consumed. To minimize system resource consumption, configure the maximum number of routes that the BGP device can accept from a BGP-LS peer.

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 BGP 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 BGP 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.

1.1.11.30 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.

  NOTE

  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.

 Copyright © Huawei Technologies Co., Ltd.

- 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.

- BGP4+ RPKI

  Resource Public Key Infrastructure (RPKI) improves BGP4+ security by validating the origin ASs of BGP4+ routes.

NOTE

GTSM supports only unicast addresses. Therefore, configure GTSM on all the Routers configured with routing protocols.

Pre-configuration Tasks

Before configuring BGP4+ security, **configure basic BGP4+ functions**.

1.1.11.30.1 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.

 Copyright © Huawei Technologies Co., Ltd.

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 NE40E-M2 series Configuration Guide > Security*.

 Copyright © Huawei Technologies Co., Ltd.

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** *sndId* **receive-id** *rcvId* 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** *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.
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.

    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.

1.1.11.30.2 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.

1.1.11.30.3 Configuring ROA

Resource Public Key Infrastructure (RPKI) ensures BGP4+ security by validating the origin ASs of BGP4+ 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**.

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:

  a. Run **system-view**

  The system view is displayed.

  b. Run **rpki**

  RPKI is started, and the RPKI view is displayed.

  c. Run **session** *ipv6-address*

  An IPv6 address 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 between the device and the RPKI server.

  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 **rpki-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 **rpki-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* | *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

After RPKI session configurations are changed, run the **reset rpki 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 **rpki**

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**

Exit the RPKI origin-validation view.

f. Run **quit**

Exit the RPKI view.

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** { *peerIpv6Addr* | *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 rpki session** *ipv6-address* **verbose** command to check RPKI session configurations.
- Run the **display rpki table** command to check ROA information.

1.1.11.30.4 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**.

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.

- 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.

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 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** *ipv6-address* **verbose** command to check RPKI session configurations.

1.1.11.31 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 1-168**, 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 1-168 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.

1.1.11.31.1 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

1.1.11.31.2 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 NE40E 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

1.1.11.31.3 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** *peerIpv4Addr* **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 in which MPLS TTLs in 6PE route labels are processed is configured.

Step 12 Run **commit**

The configuration is committed.

----End

1.1.11.31.4 (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

1.1.11.31.5 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, RIPng, or IS-IS. You can select one of them as required. For details, see the configuration of each routing protocol in the *HUAWEI NE40E-M2 seriesUniversal Service Router Configuration Guide - IP Routing*.

1.1.11.31.6 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

1.1.11.32 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 1-169**, PE1, P, and PE2 run an IGP for interworking, an LDP LSP or MPLS TE tunnel is established between PE1 and PE2, and an 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 1-169 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.

1.1.11.33 Maintaining BGP4+

Maintaining BGP4+ involves resetting BGP4+ connections, clearing BGP4+ statistics, and debugging BGP4+.

1.1.11.33.1 Resetting BGP4+ Connections

Resetting BGP4+ connections interrupts peer relationships.

Context

NOTICE

BGP4+ peer relationships between Routers are deleted if you reset BGP4+ connections using the **reset bgp ipv6** command. Exercise caution when resetting BGP4+ connections.

When a BGP4+ routing policy (the Router does not support the router-fresh 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.
- Run the **reset bgp ipv6 internal** command to reset all IBGP connections.

----End

 Copyright © Huawei Technologies Co., Ltd.

1.1.11.33.2 Clearing BGP4+ Statistics

Clearing BGP4+ statistics involves clearing route flapping statistics and route dampening statistics.

Context

NOTICE

BGP4+ statistics cannot be restored after they are cleared. Exercise caution when using reset commands.

Procedure

- 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.
- Run the **reset bgp ipv6** *ipv6-address* **flap-info** command in the user view to clear route flapping statistics on a specified peer.
- Run the **reset bgp ipv6 dampening** [*ipv6-address prefix-length*] command in the user view to clear route dampening statistics and release suppressed routes.

----End

1.1.11.33.3 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 configuration may cause the specified path attributes to be dropped or the routes carrying the specified path attributes to be withdrawn. Therefore, exercise caution when you perform this configuration.

The **peer path-attribute-treat** command configuration takes effect immediately for the routes received after the command is run. However, for the routes received before the command is run, the configuration can take effect only after the **refresh bgp** command is run.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp** *as-number*

BGP is started (with the local AS number specified), and the BGP view is displayed.

Step 3 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 4 Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

Step 5 Run the **peer** { *peerIpv4Addr* | *peerIpv6Addr* } **path-attribute-treat attribute-id** { *id* [**to** *id2*] } &<1-255> { **discard** | **withdraw** | **treat-as-unknown** } command to configure a mode for the device to process specified attributes. Alternatively, run the **peer** { *peerIpv4Addr* | *peerIpv6Addr* } **treat-with-error attribute-id** *id* **accept-zero-value** command to configure a mode for the device to process incorrect path attributes.

BGP Update messages contain various path attributes. If the local device receives any Update message with an incorrect format, BGP session flapping may occur. To enhance reliability, you can perform this step to configure a processing mode for specified BGP path attributes or incorrect path attributes.

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

The **treat-with-error** parameter specifies a processing mode for incorrect path attributes. The mode can be as follows:

- Accepting the path attributes with a value of 0.

Step 6 Run **commit**

The configuration is committed.

----End

1.1.11.33.4 Setting a Priority That Determines the Disconnection Order of a BGP4+ Peer Relationship If Memory Overload Occurs

You can set a priority that determines the disconnection order of a BGP4+ peer relationship upon memory overload. If the system memory usage exceeds the alarm threshold and the BGP4+ memory usage is excessively high, such configurations allow BGP4+ peer relationships to be disconnected in order of priority. This prevents BGP4+ from exhausting the memory.

Usage Scenario

If BGP4+ consumes excessive memory resources, a board reset may occur. To prevent this issue, you can set a priority that determines the disconnection order

of a BGP4+ peer relationship upon memory overload. Memory overload here means that the system memory usage exceeds the alarm threshold and the BGP4+ memory usage is excessively high. If this occurs, the BGP4+ peer relationships are disconnected in order of priority to release memory. This ensures that the system operates as normal. A lower priority indicates that the associated peer relationship will be disconnected first.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp** *as-number*

BGP is started (with the local AS number specified), and the BGP view is displayed.

Step 3 Run **peer** *ipv6-address* **as-number** *as-number*

The IPv6 address of a peer and the number of the AS where the peer resides are specified.

Step 4 Run **ipv6-family unicast**

The BGP-IPv6 unicast address family view is displayed.

Step 5 Run **peer** *peerIpv6Addr* **memory-priority** *priority*

A priority is set, which determines the disconnection order of the specified BGP4+ peer relationship if memory overload occurs.

Step 6 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

- Run the **display bgp peer verbose** command to check the priority that determines the disconnection order of the specified BGP4+ peer relationship if memory overload occurs.

1.1.11.33.5 Enabling BGP4+ to Discard New Routes in the Case of a Memory Exception

After BGP4+ is enabled to discard new routes in the case of a memory exception, BGP4+ will discard the new routes received from peers in the case of a memory exception.

Usage Scenario

If BGP4+ keeps accepting new routes in the case of a memory exception, the device memory may be exhausted. As a result, the involved process or board is reset. To prevent this problem, you can enable BGP4+ to discard new routes in the case of a memory exception.

 Copyright © Huawei Technologies Co., Ltd.

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **bgp memory-usage exception-threshold discard new-route**

BGP4+ is enabled to discard new routes if a memory exception occurs.

Step 3 Run **commit**

The configuration is committed.

----End

Verifying the Configuration

- Run the **display bgp routing-table** command to check changes in the BGP4+ routing table.

1.1.11.33.6 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.

1.1.11.33.7 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.

  NOTE

  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.

1.1.11.33.8 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.

NOTE

If the function becomes abnormal or affects other services, you can run the **whitelist session-car rpki 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 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.

1.1.11.34 Configuration Examples for BGP4+

This chapter provides several BGP4+ configuration examples.

1.1.11.34.1 Example for Configuring Basic BGP4+ Functions

Before building BGP4+ networks, you need to configure basic BGP4+ functions.

Networking Requirements

In **Figure 1-170**, 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 1-170 Configuring basic BGP4+ functions

NOTE

Interfaces 1 through 4 in this example are GE 0/1/0, GE 0/2/0, GE 0/3/0, GE 0/1/0, 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      :                               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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:8::1/64
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:9:1::1/64
```

```
#
interface GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:10::1/64
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:9:2::1/64
#
interface GigabitEthernet0/3/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:9:1::2/64
#
interface GigabitEthernet0/2/0
```

```
 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
```

1.1.11.34.2 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 1-171**, 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 an Update packet from Device B, Device C reflects it to Device D. Similarly, after receiving an Update packet from Device D, Device C reflects it to Device B.

Figure 1-171 Configuring BGP4+ route reflection

NOTE

Interfaces 1 and 2 in this example are GE 0/1/0 and GE 0/2/0, 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
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::1/64
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:101::2/96
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:102::1/96
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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
```

1.1.11.34.3 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 1-172**, 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 1-172 Networking diagram for configuring BGP4+ keychain

NOTE

Interface1 in this example represents GE0/1/0.

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 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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 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
```

1.1.11.34.4 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 1-173**, 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 1-173 Configuring BFD for BGP4+

NOTE

Interfaces 1 through 3 in this example are GE 0/1/0, GE 0/2/0, and GE 0/3/0, 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      : GigabitEthernet0/1/0
  Session-State  : Up
  Wtr-interval(m): 0
-------------------------------------------------------------------------------
```

Step 6 Verify the configuration.

Run the **shutdown** command on GE 0/2/0 of DeviceB to simulate a fault in the primary link.

```
[~DeviceB] interface gigabitethernet 0/2/0
[~DeviceB-Gigabitethernet0/2/0] 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:8::1/64
#
interface GigabitEthernet0/2/0
```

```
 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 GigabitEthernet0/2/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:8::2/64
#
interface GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:9::1:1/64
#
interface GigabitEthernet0/3/0
 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:: 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 GigabitEthernet0/1/0
 undo shutdown
```

```
 ipv6 enable
 ipv6 address 2001:db8:9::1:2/64
#
interface GigabitEthernet0/2/0
 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
```

1.1.11.34.5 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 1-174**, 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 1-174 Configuring BGP4+ 6PE

NOTE

Interfaces 1 through 4 in this example represent GE 0/1/0, GE 0/2/0, 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 gigabitethernet0/2/0
[*PE1-GigabitEthernet0/2/0] mpls
[*PE1-GigabitEthernet0/2/0] mpls ldp
[*PE1-GigabitEthernet0/2/0] 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 gigabitethernet0/2/0
[*PE2-GigabitEthernet0/2/0] mpls
[*PE2-GigabitEthernet0/2/0] mpls ldp
[*PE2-GigabitEthernet0/2/0] 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
```

 Copyright © Huawei Technologies Co., Ltd.

```
[*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::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    : GigabitEthernet0/1/0         Flags        : D

 Destination  : 2001:db8:1::1                PrefixLength : 128
 NextHop      : ::1                          Preference   : 0
 Cost         : 0                            Protocol     : Direct
 RelayNextHop : ::                           TunnelID     : 0x0
 Interface    : GigabitEthernet0/1/0         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    : GigabitEthernet0/1/0         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 GigabitEthernet0/1/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:1::2/64
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 undo shutdown
 ipv6 enable
 ipv6 address 2001:db8:2::2/64
#
interface GigabitEthernet0/2/0
 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 GigabitEthernet0/1/0
 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
```

1.1.12 Routing Policy Configuration

Routing policies are applied to routing information to change the path through which network traffic passes.

1.1.12.1 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 1-49** shows the differences between them.

Table 1-49 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.
Is configured using the **route-policy** command.	Is configured using the **policy-based-route** command.

 Copyright © Huawei Technologies Co., Ltd.

1.1.12.2 Configuration Precautions for Routing Policy

Feature Requirements

Table 1-50 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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
If the route-policy nonexistent-config-check disable command is not run, nonexistent route-policies cannot be applied to services.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.12.3 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 the routes, you need to configure an IP prefix list.

Each route contains an IP prefix. Therefore, IP prefix lists can be applied on various networks to filter routes.

Before applying a routing policy, you need to set the matching rules (filters). Similar to an ACL, an IP prefix list is flexible in filtering routes. IP prefix lists filter routes based on the destination addresses of the routes.

Pre-configuration Tasks

None

1.1.12.3.1 Configuring an IPv4 Prefix List

An IP prefix list filters routes based on IP address prefixes. Each IP prefix in an IP prefix list contains an IP address and a mask length.

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** } *ip-address mask-length* [**match-network**] [**greater-equal** *greater-equal-value*] [**less-equal** *less-equal-value*]

An IPv4 prefix list is configured.

The range of the mask length must meet the following requirements: *mask-length* <= *greater-equal-value* <= *less-equal-value* <= 32. If only **greater-equal** is specified, the range of the IP prefix is [*greater-equal-value*, 32]. If only **less-equal** is specified, the range of the IP prefix 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.

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.

On the NE40E, the routes that fail to match an IPv4 prefix list are denied by default. If all entries are set to **deny** mode, all routes are denied by the IP prefix list. Therefore, you need to define an entry **permit 0.0.0.0 0 less-equal 32** following the entries in **deny** mode so that other IPv4 routes will not be filtered out by the IP prefix list.

NOTE

If more than one IP prefix entry is defined, at least one entry needs to be set in **permit** mode.

Step 3 Run **commit**

The configuration is committed.

----End

1.1.12.3.2 Configuring an IPv6 Prefix List

An IPv6 prefix list filters routes based on IPv6 address prefixes. Each IPv6 prefix in an IPv6 prefix list contains an IPv6 address and a prefix length.

Context

Perform the following steps on the device where an IPv6 prefix list needs to be applied:

Procedure

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **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*]

An IPv6 prefix list is configured.

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.

On the NE40E, all unmatched routes are denied by the prefix list by default. If all entries in a prefix list are in **deny** mode, all routes will be denied. 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 entry is defined for a prefix list, at least one of the entries must be set in **permit** mode.

Step 3 Run **commit**

The configuration is committed.

----End

1.1.12.3.3 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

1.1.12.4 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 can be configured for both IPv4 routes and IPv6 routes, and are 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-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.

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.

1.1.12.5 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.

1.1.12.6 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.

NOTE

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.

1.1.12.7 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 NE40E 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 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.

1.1.12.8 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 out 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.

The matching condition of an extcommunity filter can be specified using an extcommunity ID or a regular expression.

NOTE

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.

Multiple entries (or rules) can be defined in an extended community 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 extcommunity filters.

1.1.12.9 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.

1.1.12.10 Configuring a Route-Policy

Each node of a route-policy can comprise a set of **if-match**, **goto next-node** and **apply** 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 attributes of routes, and to change the attributes when the matching rules are met. The matching rules of a route-policy

can use other filters, such as ACLs, IP prefix lists, AS_Path filters, community filters, extended community filters, and RD filters.

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 NE40E-M2 series Universal Service 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.

1.1.12.10.1 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 NE40E 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 **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 4 Run **commit**

The configuration is committed.

----End

1.1.12.10.2 (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.

- To configure a rule to match routes against a basic ACL, perform the following steps:

  a. Run **if-match acl** { *acl-number* | *acl-name* }

  A matching rule based on a 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 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** | **ebgp** } 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 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 rpki 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 match non-optimal BGP routes, run the **if-match route-state bgp-not-best** 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-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 rpki 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 route-state bgp-not-best**

If no **if-match** clause is specified, all routes will match the route-policy node.

You are not advised 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

1.1.12.10.3 (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 **apply as-path none overwrite** command.
- 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** } or **apply extcommunity bandwidth aggregate** [**limit** *bandwidth-value*] 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 a specified MPLS label value 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* } | **igp** | **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 an IP gateway address for matched routes, run the **apply gateway-ip** { **origin-nexthop** | *ip-address* | **none** } command.
- To set an IPv6 gateway address for matched routes, run the **apply ipv6 gateway-ip** { **origin-nexthop** | *ipv6-address* | **none** } command.
- To discard the entropy label of routes, run the **apply entropy-label none** command.
- To stitch different types of 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

1.1.12.10.4 (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

1.1.12.10.5 Applying a Route-Policy

A route-policy takes effect only when it is applied to a routing protocol.

Context

A route-policy can be applied to direct, static, RIP/RIPng, IS-IS, OSPF/OSPFv3, BGP/BGP4+, and BGP/MPLS IP VPN routes. In addition, a route-policy can be applied to an FRR scenario. Perform 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 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 routes.

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 1-51**.

Table 1-51 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 **route-policy** *route-policy-name* parameter)	**arp vlink-direct-route advertise** [**route-policy** *route-policy-name*]	**1.1.2.16 Configuring the Advertisement of Public Network IPv4 ARP Vlink Direct Routes**

Objectives	Command	Reference
To configure a device to advertise Neighbor Discovery Protocol (NDP) Vlink direct routes that match a route-policy (specified by the **route-policy** *route-policy-name* parameter) on the public network	**ipv6 nd vlink-direct-route advertise** [**route-policy** *route-policy-name*]	**1.1.2.17 Configuring the Advertisement of IPv6 NDP Vlink Direct Routes on the Public Network**
To apply a route-policy to direct routes on the public network **NOTE** 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.	**ip direct-routing-table route-policy** *route-policy-name*	-

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 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 1-52**.

Table 1-52 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 **route-policy** *route-policy-name* parameter) are met	**default-route originate** [**cost** *cost* | **tag** *tag* | **route-policy** *route-policy-name* [**avoid-learning**]] *	**1.1.7.6.2 Configuring RIP to Advertise Default Routes**
To configure RIP to import the BGP routes that match a route-policy (specified by the **route-policy** *route-policy-name* parameter)	**import-route bgp** [**permit-ibgp**] [**cost** { *cost* | **transparent** } | **route-policy** *route-policy-name*] *	**1.1.7.6.1 Configuring RIP to Import External Routes**

Objectives	Command	Reference
To configure RIP to import the routes from another routing protocol that match a route-policy (specified by the **route-policy** *route-policy-name* parameter)	**import-route** { **static** | **direct** | **unr** | { **isis** | **ospf** | **rip** } [*process-id*] } [**cost** *cost* | **route-policy** *route-policy-name*] *	1.1.7.6.1 Configuring RIP to Import External Routes
To set a priority for RIP routes that match a route-policy (specified by the **route-policy** *route-policy-name* parameter)	**preference** { *preference* | **route-policy** *route-policy-name* } *	1.1.7.3.4 (Optional) Configuring a RIP Preference

d. Run **commit**

The configuration is committed.

- Apply a route-policy to RIPng routes.

a. Run **system-view**

The system view is displayed.

b. Run **ripng** [*process-id*]

A RIPng process is enabled, and the RIPng view is displayed.

c. Perform the following operations as required to apply a route-policy to RIPng routes. For details, see Table 1-53.

Table 1-53 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 **route-policy** *route-policy-name* parameter)	**import-route** { **static** | **direct** | **bgp** [**permit-ibgp**] | **unr** | { **isis** | **ospfv3** | **ripng** } [*process-id*] } [**cost** *cost* | **route-policy** *route-policy-name*] *	1.1.8.7.1 Configuring RIPng to Import External Routes

Objectives	Command	Reference
To set a priority for RIPng routes that match a route-policy (specified by the **route-policy** *route-policy-name* parameter)	**preference** { *preference* | **route-policy** *route-policy-name* } *	**1.1.8.3.3 (Optional) Configuring the RIPng Priority**

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 1-54**.

Table 1-54 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 **route-policy** *route-policy-name* 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.)	**default-route-advertise route-policy** *route-policy-name* [**cost** *cost* | **tag** *tag* | [**level-1** | **level-1-2** | **level-2**]] * [**avoid-learning**]	**1.1.9.4.5 Configuring IS-IS to Generate IPv4 Default Routes**
To configure IS-IS to advertise the routes that are imported from another routing protocol and match a route-policy	**filter-policy route-policy** *route-policy-name* **export** [*protocol* [*process-id*]]	-
To configure IS-IS to accept the routes that match a route-policy	**filter-policy route-policy** *route-policy-name* **import**	**1.1.9.5.2 Configuring IPv4 IS-IS to Import External Routes**

 Copyright © Huawei Technologies Co., Ltd.

Objectives	Command	Reference
To configure IS-IS to import the routes from another routing protocol that match a route-policy	**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**]] * **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*] * **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**] * **import-route** { **ospf** | **isis** } [*process-id*] **inherit-cost** [{ **level-1** | **level-2** | **level-1-2** } | **tag** *tag* | **route-policy** *route-policy-name* | **no-sid**] *	**1.1.9.5.2 Configuring IPv4 IS-IS to Import External Routes**
To allow Level-1 routes to leak to a Level-2 area	**import-route isis level-1 into level-2** [**filter-policy route-policy** *route-policy-name* | **direct allow-filter-policy** | **tag** *tag* | **no-sid**] *	**1.1.9.4.1 Configuring IPv4 IS-IS Route Leaking**
To allow Level-2 routes to leak to a Level-1 area	**import-route isis level-2 into level-1** [**filter-policy route-policy** *route-policy-name* | **direct allow-filter-policy** | **tag** *tag* | **no-sid**] *	**1.1.9.4.1 Configuring IPv4 IS-IS Route Leaking**

 Copyright © Huawei Technologies Co., Ltd.

Objectives	Command	Reference
To configure a priority for the IS-IS routes that match a route-policy	**preference** { **route-policy** *route-policy-name* | *preference* } *	1.1.9.5.1 Configuring a Preference Value for IPv4 IS-IS

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 1.1.9.15 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 1-55.

Table 1-55 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 **route-policy** *route-policy-name* parameter) exist in the routing table of a border device	**ipv6 default-route-advertise route-policy** *route-policy-name* [**cost** *cost* | **tag** *tag* | [**level-1** | **level-2** | **level-1-2**]] * [**avoid-learning**| **learning-avoid-loop**]	**1.1.9.17.5 Configuring IS-IS to Generate IPv6 Default Routes**
To configure IS-IS to advertise the routes that are imported from another routing protocol and match a route-policy	**ipv6 filter-policy route-policy** *route-policy-name* **export** [*protocol* [*process-id*]]	-
To configure IS-IS to accept the IPv6 routes that match a route-policy	**ipv6 filter-policy route-policy** *route-policy-name* **import**	**1.1.9.18.2 Configuring IPv6 IS-IS to Import External Routes**

 Copyright © Huawei Technologies Co., Ltd.

Objectives	Command	Reference
To configure IS-IS to import the routes from another routing protocol that match a route-policy	**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**]] * **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**]] *	**1.1.9.18.2 Configuring IPv6 IS-IS to Import External Routes**
To configure Level-1 IPv6 routes that match a route-policy to leak to a Level-2 area	**ipv6 import-route isis level-1 into level-2** [**filter-policy route-policy** *route-policy-name* | **direct allow-filter-policy** | **tag** *tag*] *	**1.1.9.17.1 Configuring IPv6 IS-IS Route Leaking**
To configure Level-2 IPv6 routes that match a route-policy to leak to a Level-1 area	**ipv6 import-route isis level-2 into level-1** [**filter-policy route-policy** *route-policy-name* | **direct allow-filter-policy** | **tag** *tag*] *	**1.1.9.17.1 Configuring IPv6 IS-IS Route Leaking**
To configure a priority for the IS-IS routes that match a route-policy	**ipv6 preference** { **route-policy** *route-policy-name* | *preference* } *	**1.1.9.18.1 Configuring a Priority Value for IPv6 IS-IS**

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 **1.1.9.29.1 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 1-56**.

Table 1-56 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 **route-policy** *route-policy-name* parameter)	**default-route-advertise** [[**always** | **permit-calculate-other**] | **cost** *cost* | **type** *type* | **route-policy** *route-policy-name* | **distribute-delay** *delay-time*] * **default-route-advertise** [**permit-calculate-other** | **cost** *cost* | **type** *type* | **route-policy** *route-policy-name* | **distribute-delay** *delay-time* | **permit-ibgp**] * **default-route-advertise summary cost** *cost*	**1.1.5.6.2 Configuring OSPF to Import a Default Route**
To configure OSPF to accept the routes that match a route-policy	**filter-policy route-policy** *route-policy-name* [**secondary**] **import**	**1.1.5.6.6 Configuring OSPF to Filter Received Routes**
To configure OSPF to import the routes that match a route-policy	**filter-policy route-policy** *route-policy-name* **export** [*protocol* [*process-id*]]	**1.1.5.6.7 Configuring OSPF to Filter the Routes to Be Advertised**

Objectives	Command	Reference
To configure OSPF to import routes from another protocol	**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*] * }	**1.1.5.6.1 Configuring OSPF to Import External Routes**
To configure a route-policy for OSPF local MT so that only the routes that match the route-policy (specified by the **route-policy** *route-policy-name* parameter) are added to the MIGP routing table	**local-mt filter-policy route-policy** *route-policy-name*	-
To configure a priority for OSPF routes that match a route-policy	**preference** [**ase** | **inter** | **intra**] { *preference* | **route-policy** *route-policy-name* } *	**1.1.5.3.5 (Optional) Setting the OSPF Priority**

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:

- 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.
- 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 **1.1.5.19 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 1-57**.

Table 1-57 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 **route-policy** *route-policy-name* parameter)	**default-route-advertise** [**always** | **permit-calculate-other** | **cost** *cost* | **type** *type* | **tag** *tag* | **distribute-delay** *delay* | **route-policy** *route-policy-name*] *	**1.1.6.6.4 Configuring OSPFv3 to Filter the Routes to Be Advertised**

Objectives	Command	Reference
To configure OSPFv3 to accept the routes that match a route-policy	**filter-policy route-policy** *route-policy-name* [**secondary**] **import**	**1.1.6.6.3 Configuring OSPFv3 to Filter Received Routes**
To configure OSPFv3 to import the routes that match a route-policy	**filter-policy route-policy** *route-policy-name* **export** [*protocol* [*process-id*]]	**1.1.6.6.4 Configuring OSPFv3 to Filter the Routes to Be Advertised**
To configure OSPFv3 to import routes from another protocol	**import-route** { **bgp** [**permit-ibgp**] | **direct** | **unr** | **static** | **isis** [*process-id*] | **ripng** [*process-id*] | **ospfv3** [*process-id*] } [**cost** *cost* | **tag** *tag* | **type** *type* | **route-policy** *route-policy-name*] * }	**1.1.6.6.1 Configuring OSPFv3 to Import External Routes**
To configure a priority for OSPF routes that match a route-policy	**preference** [**ase**] { *preference* | **route-policy** *route-policy-name* } *	-

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:

- 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.
- 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 **1.1.6.6.5 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 **1.1.6.15 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 1-58**.

Table 1-58 Applying a route-policy to BGP routes

Objectives	Command	Reference
To configure a summary BGP route	**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*] *	**1.1.10.8 Configuring BGP Route Summarization**
To configure BGP route dampening	**dampening** [*half-life-reach reuse suppress ceiling* | **route-policy** *route-policy-name*] *[**update-standard**]	**1.1.10.41 Configuring BGP Route Dampening**
To configure BGP to import routes from another routing protocol	**import-route** *protocol* [*process-id*] [**med** *med* | **route-policy** *route-policy-name*] *[**non-relay-tunnel**]	**1.1.10.3.3 Configuring BGP to Import Routes**

Objectives	Command	Reference
To configure BGP to import local routes	**network** *ipv4-address* [*mask* | *mask-length*] [**route-policy** *route-policy-name*] [**non-relay-tunnel**]	**1.1.10.3.3 Configuring BGP to Import Routes**
To enable BGP route recursion based on a route-policy	**nexthop recursive-lookup route-policy** *route-policy-name*	**1.1.10.5.5 Configuring the Next_Hop Attribute**
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 **route-policy** *route-policy-name* parameter) to modify the attributes of the default route	**peer** { *group-name* | *ipv4-address* } **default-route-advertise** [**route-policy** *route-policy-name*] [**conditional-route-match-all** { *ipv4-address1* { *mask1* | *mask-length1* } } &<1-4> | **conditional-route-match-any** { *ipv4-address2* { *mask2* | *mask-length2* } } &<1-4>]	**1.1.10.18 Configuring a BGP Device to Send a Default Route to Its Peer**
To configure BGP to ignore bit error detection results when an export route-policy is used. **NOTE** 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.	**peer** { *group-name* | *ipv4-address* } **route-policy** *route-policy-name* **export ignore-bit-error**	-

 Copyright © Huawei Technologies Co., Ltd.

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	**peer** { *group-name* | *ipv4-address* } **route-policy** *route-policy-name* { **import** | **export** }	**1.1.10.6.2 Applying a Policy to BGP Route Advertisement** **1.1.10.6.3 Applying a Policy to BGP Route Acceptance**
To use a route-policy to set a BGP priority	**preference route-policy** *route-policy-name*	**1.1.10.5.1 Setting the BGP Preference**
To prevent the BGP routes that match a route-policy from being added to the IP routing table	**routing-table rib-only** [**route-policy** *route-policy-name*]	**1.1.10.11.4 (Optional) Preventing a Device from Adding BGP Routes to the IP Routing Table**

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 1-59**.

Table 1-59 Applying a route-policy to BGP4+ routes

Objectives	Command	Reference
To configure a summary BGP4+ route	**aggregate** *ipv6-address prefix-length* [**as-set** | **attribute-policy** *route-policy-name1* | **detail-suppressed** | **origin-policy** *route-policy-name2* | **suppress-policy** *route-policy-name3*] *	**1.1.11.6.1 Configuring BGP4+ Route Summarization**
To configure BGP4+ route dampening and apply dampening parameters to the routes that match a route-policy (specified by the **route-policy** *route-policy-name* parameter) so that specified-requirement dampening can be performed by BGP based on the applied route-policy	**dampening** [*half-life-reach reuse suppress ceiling* | **route-policy** *route-policy-name*] *	**1.1.11.26 Configuring BGP4+ Route Dampening**
To configure BGP4+ to import the routes from another routing protocol that match a route-policy	**import-route** *protocol* [*process-id*] [**med** *med* | **route-policy** *route-policy-name*] *	**1.1.11.5.1 Configuring BGP4+ to Import Routes**
To configure BGP4+ to advertise local routes	**network** *ipv6-address prefix-length* [**route-policy** *route-policy-name*]	**1.1.11.5.1 Configuring BGP4+ to Import Routes**
To enable BGP4+ route recursion based on a route-policy	**nexthop recursive-lookup route-policy** *route-policy-name*	**1.1.11.7.5 Setting the Next_Hop Attribute**

 Copyright © Huawei Technologies Co., Ltd.

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 **route-policy** *route-policy-name* parameter) to modify the attributes of the default route	**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* } } &<1-4>	**1.1.11.6.2 Configuring BGP4+ to Advertise Default Routes to Peers or peer groups**
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	**peer** { *group-name* | *ipv6-address* } **route-policy** *route-policy-name* { **import** | **export** }	**1.1.11.8.3 Applying a Policy to BGP4+ Route Advertisement** **1.1.11.8.4 Configuring a Policy for Receiving BGP4+ Routes**
To use a route-policy to set a BGP4+ priority	**preference route-policy** *route-policy-name*	**1.1.11.7.1 Setting a BGP4+ Preference**
To prevent the BGP4+ routes that match a route-policy from being added to the IPv6 routing table	**routing-table rib-only** [**route-policy** *route-policy-name*]	**1.1.11.15.4 (Optional) Preventing BGP4+ from Adding Routes to the IP Routing Table**

e. Run **commit**

The configuration is committed.

- Apply a route-policy to BGP/MPLS IP VPN.
  - 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 in the VPN instance view, perform one of the following operations as required:

○ 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.

○ 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.

○ 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 Configure a direct route between a PE and a CE.

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 **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 Configure a direct route between a PE and a CE.

v. Run the **commit** command to commit the configuration.

----End

1.1.12.10.6 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

1.1.12.11 Applying Filters to Received Routes

By applying filters of routing policies to routing protocols, you can filter the received routes.

Usage Scenario

When exchanging routes on a network, devices need to accept only required routes. In this situation, you can define a filter (such as an IP prefix list, ACL, or a route-policy) for a routing policy, apply the filter to the routing protocol, and run the **filter-policy** command specified with the filter in the related protocol view to filter received routes. Then, devices accept only the routes that meet the matching rules.

The function of the **filter-policy import** 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 those to be advertised to neighbors.

- Link-state protocol

  A link-state protocol generates routes based on the link state database (LSDB). The **filter-policy** command does not affect any Link State Advertisement (LSA) or LSDB.

  After routes are received, the **filter-policy** command determines which routes to be added from the protocol routing table to the local core routing table. Therefore, this command takes effect on the local core routing table rather than the protocol routing table.

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 received routes, complete the following tasks:

- **Configure an IP prefix list**.
- Configure an ACL.
- **Configure a route-policy**.

1.1.12.11.1 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

1.1.12.11.2 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

1.1.12.11.3 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

1.1.12.11.4 Verifying the Configuration of Applying Filters to Received Routes

After applying filters to the received routes, verify 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 to view that the routes that meet the matching rules set on the neighbor are filtered or the actions defined by **apply** clauses are performed on these routes.

----End

1.1.12.12 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**.

1.1.12.12.1 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

1.1.12.12.2 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 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* } **export** [**direct** | **static** | **unr** | **bgp** | { **rip** | **isis** | **ospf** } [*process-id*]]

An export policy that is based on the basic ACL is configured for OSPF to filter 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*]]

  An export policy that is based on the IP prefix list is configured for OSPF to filter 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*]]

  An export policy that is based on the route-policy is configured for OSPF to filter imported routes to be advertised.

Step 4 Run **commit**

The configuration is committed.

----End

1.1.12.12.3 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

1.1.12.12.4 Verifying the Configuration of Applying Filters to the Advertised Routes

After applying filters to the advertised routes, check information about the routing table of each protocol.

Prerequisites

Filters have been applied to the advertised 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 neighbor to view that the routes that meet the matching rules set on the neighbor are filtered or the actions defined by **apply** clauses are performed on these routes.

----End

1.1.12.13 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**.

1.1.12.13.1 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

1.1.12.13.2 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*] *

     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* } *

     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

1.1.12.13.3 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

1.1.12.13.4 Verifying the Configuration of Applying Filters to the Imported Routes

After applying filters to the imported routes, verify 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 that meet the matching rules set on the neighbor are filtered or the actions defined by **apply** clauses are performed on these routes.

----End

1.1.12.14 Configuration Examples for Routing Policies

This section provides some routing policy configuration examples.

1.1.12.14.1 Example for Filtering the Received Routes and Those to Be Advertised

Filters can be applied to the received routes and those to be advertised based on networking requirements.

Networking Requirements

In Figure 1-175, on the network running OSPF, Device A receives routes from the Internet and provides some Internet routes for Device B. It is required that Device A provide only 172.16.17.0/24, 172.16.18.0/24, and 172.16.19.0/24 for Device B, Device C receive only 172.16.18.0/24, and that Device D receive all the routes provided by Device B.

Figure 1-175 Filtering the received routes and those to be advertised

NOTE

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

Precautions

When filtering received routes and those to be advertised, note the following rules:

- When configuring an IP prefix list, specify the IP prefix range as required.
- The IP prefix list name is case sensitive.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic OSPF functions on Device A, Device B, Device C, and Device D.
2. Configure static routes on Device A and import these routes into OSPF.
3. Configure a policy to filter routes to be advertised on Device A and then check the routing table on Device B.
4. Configure a policy to filter received routes on Device C and then check the routing table on Device C.

Data Preparation

To complete the configuration, you need the following data:

- Five static routes imported by Device A
- Area 0 (backbone area), in which Device A, Device B, Device C, and Device D are located

- Name of the IP prefix list and routes to be filtered

Procedure

Step 1 Configure IP addresses for interfaces. For configuration details, see **Configuration Files** in this section.

Step 2 Configure OSPF.

Configure Device A.

```
[~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 Device B.

```
[~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 Device C.

```
[~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 Device D.

```
[~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 Device A, and import these routes into OSPF.

```
[~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
```

Display the IP routing table on Device B. The following command output shows that the five static routes have been imported into 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
  172.16.16.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
  172.16.17.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
  172.16.18.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
  172.16.19.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
  172.16.20.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
   192.168.1.0/24  Direct 0    0          D  192.168.1.2     GigabitEthernet0/1/0
   192.168.1.1/32  Direct 0    0          D  192.168.1.1     GigabitEthernet0/1/0
   192.168.1.2/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/1/0
 192.168.1.255/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/1/0
   192.168.2.0/24  Direct 0    0          D  192.168.2.1     GigabitEthernet0/3/0
   192.168.2.1/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/3/0
   192.168.2.2/32  Direct 0    0          D  192.168.2.2     GigabitEthernet0/3/0
 192.168.2.255/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/3/0
   192.168.3.0/24  Direct 0    0          D  192.168.3.1     GigabitEthernet0/2/0
   192.168.3.1/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/2/0
   192.168.3.2/32  Direct 0    0          D  192.168.3.2     GigabitEthernet0/2/0
 192.168.3.255/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/2/0
```

Step 4 Configure a policy to filter routes to be advertised.

Configure the IP prefix list named **a2b** on Device A.

```
[~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 a policy on Device A and use the IP prefix list named **a2b** to filter routes.

```
[~DeviceA] ospf
[*DeviceA-ospf-1] filter-policy ip-prefix a2b export static
[*DeviceA-ospf-1] commit
[~DeviceA-ospf-1] quit
```

Display the IP routing table on Device B. The following command output shows that Device B has received only three routes specified in the IP prefix list named **a2b**.

```
[~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 : 20        Routes : 20

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
  172.16.17.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
  172.16.18.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
  172.16.19.0/24  O_ASE  150  1           D  192.168.1.1     GigabitEthernet0/1/0
   192.168.1.0/24  Direct 0    0          D  192.168.1.2     GigabitEthernet0/1/0
   192.168.1.1/32  Direct 0    0          D  192.168.1.1     GigabitEthernet0/1/0
   192.168.1.2/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/1/0
 192.168.1.255/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/1/0
   192.168.2.0/24  Direct 0    0          D  192.168.2.1     GigabitEthernet0/3/0
   192.168.2.1/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/3/0
   192.168.2.2/32  Direct 0    0          D  192.168.2.2     GigabitEthernet0/3/0
 192.168.2.255/32  Direct 0    0          D  127.0.0.1       GigabitEthernet0/3/0
   192.168.3.0/24  Direct 0    0          D  192.168.3.1     GigabitEthernet0/2/0
```

```
   192.168.3.1/32  Direct 0    0           D  127.0.0.1       GigabitEthernet0/2/0
   192.168.3.2/32  Direct 0    0           D  192.168.3.2     GigabitEthernet0/2/0
 192.168.3.255/32  Direct 0    0           D  127.0.0.1       GigabitEthernet0/2/0
```

Step 5 Configure a policy to filter received routes.

Configure the IP prefix list named **in** on Device C.

```
[~DeviceC] ip ip-prefix in index 10 permit 172.16.18.0 24
[*DeviceC] commit
```

Configure a policy on Device C and use the IP prefix list named **in** to filter routes.

```
[~DeviceC] ospf
[*DeviceC-ospf-1] filter-policy ip-prefix in import
[*DeviceC-ospf-1] commit
```

Display the IP routing table on Device C. The following command output shows that Device C has received only one route specified in the IP prefix list named **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     GigabitEthernet0/1/0
      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     GigabitEthernet0/1/0
    192.168.1.0/24  O_ASE  10   2           D  192.168.2.1     GigabitEthernet0/1/0
    192.168.2.0/24  Direct 0    0           D  192.168.2.2     GigabitEthernet0/1/0
    192.168.2.1/32  Direct 0    0           D  192.168.2.1     GigabitEthernet0/1/0
    192.168.2.2/32  Direct 0    0           D  127.0.0.1       GigabitEthernet0/1/0
  192.168.2.255/32  Direct 0    0           D  127.0.0.1       GigabitEthernet0/1/0
    192.168.3.0/24  O_ASE  10   2           D  192.168.2.1     GigabitEthernet0/1/0
```

----End

Configuration Files

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/1/0
 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
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 192.168.3.1 255.255.255.0
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 192.168.2.1 255.255.255.0
#
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
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.2.2 255.255.255.0
#
ospf 1
 filter-policy ip-prefix in import
 area 0.0.0.0
  network 192.168.2.0 0.0.0.255
#
ip ip-prefix in index 10 permit 172.16.18.0 24
#
return
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.3.2 255.255.255.0
#
ospf 1
 area 0.0.0.0
  network 192.168.3.0 0.0.0.255
#
return
```

1.1.12.14.2 Example for Applying a Route-Policy to Imported Routes

By applying route-policies, you can control the import of routes and set attributes for imported routes.

Networking Requirements

In **Figure 1-176**, Device A and Device B exchange routing information through OSPF; Device B and Device C exchange routing information through IS-IS.

It is required that Device B import IS-IS routes to OSPF and use the route-policy to set the cost of the route to 172.16.1.0/24 to 100 and set the tag of the route to 172.16.2.0/24 to 20.

Figure 1-176 Applying a route-policy to imported routes

NOTE

Interfaces 1 through 4 in this example represent GE 0/1/0, GE 0/2/0, GE 0/3/0, and GE 0/1/1, respectively.

Precautions

When applying a route-policy to imported routes, note the following rules:

- When configuring an IP prefix list, specify the IP prefix range as required.
- The route-policy name is case sensitive.

Configuration Roadmap

The configuration roadmap is as follows:

1. Configure basic IS-IS functions on Device B and Device C.
2. Configure OSPF on Device A and Device B and import IS-IS routes.
3. Configure a routing policy on Device B, apply the route-policy when OSPF imports IS-IS routes, and check the routes.

Data Preparation

To complete the configuration, you need the following data:

- Area IDs, IS-IS levels, and system IDs of Device B and Device C
- Area 0 (backbone area), in which Device A and Device B are located
- ACL number, name of the IP prefix list, cost of the route to 172.16.1.0/24, and tag of the route to 172.16.2.0/24

Procedure

Step 1 Configure IP addresses for interfaces. For configuration details, see **Configuration Files** in this section.

Step 2 Configure IS-IS.

Configure Device C.

```
[~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 0/1/1
[*DeviceC-GigabitEthernet0/1/1] isis enable
[*DeviceC-GigabitEthernet0/1/1] quit
[*DeviceC] interface GigabitEthernet 0/1/0
[*DeviceC-GigabitEthernet0/1/0] isis enable
[*DeviceC-GigabitEthernet0/1/0] quit
[*DeviceC] interface GigabitEthernet 0/2/0
[*DeviceC-GigabitEthernet0/2/0] isis enable
[*DeviceC-GigabitEthernet0/2/0] quit
[*DeviceC] interface GigabitEthernet 0/3/0
[*DeviceC-GigabitEthernet0/3/0] isis enable
[*DeviceC-GigabitEthernet0/3/0] commit
[~DeviceC-GigabitEthernet0/3/0] 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 0/2/0
[*DeviceB-GigabitEthernet0/2/0] isis enable
[*DeviceB-GigabitEthernet0/2/0] commit
[~DeviceB-GigabitEthernet0/2/0] quit
```

Step 3 Configure OSPF and import routes.

Configure Device A and enable OSPF.

```
[~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 Device B, enable OSPF, and import IS-IS routes.

```
[~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] quit
[*DeviceB-ospf-1] import-route isis 1
[*DeviceB-ospf-1] commit
[~DeviceB-ospf-1] quit
```

Check the IP routing table on Device A to view 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 import the route to 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 import the route to 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 when routes are imported.

Configure Device B and apply the route-policy when routes are imported.

```
[~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 Device A. The command output shows that the cost of the route to 172.16.1.0/24 is 100, the tag of the route to 172.16.2.0/24 is 20, and that other route attributes 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

- Device A configuration file

```
#
 sysname DeviceA
#
interface GigabitEthernet0/1/0
 undo shutdown
```

```
 ip address 192.168.1.1 255.255.255.0
#
ospf 1
 area 0.0.0.0
  network 192.168.1.0 0.0.0.255
#
return
```

- Device B 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 GigabitEthernet0/1/0
 undo shutdown
 ip address 192.168.1.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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
```

- Device C configuration file

```
#
sysname DeviceC
#
isis 1
 is-level level-2
 network-entity 10.0000.0000.0001.00
#
interface GigabitEthernet0/1/0
 undo shutdown
 ip address 172.16.1.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/2/0
 undo shutdown
 ip address 172.16.2.1 255.255.255.0
 isis enable 1
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 172.16.3.1 255.255.255.0
 isis enable 1
#
```

```
interface GigabitEthernet0/1/1
 ip address 192.168.2.1 255.255.255.0
 isis enable 1
#
return
```

1.1.12.14.3 Example for Applying Route-Policies

By configuring route-policies, you can flexibly control the traffic on a complex network.

Networking Requirements

Figure 1-177 shows a simplified diagram of the MPLS network that carries multiple types of L3VPN services, such as multimedia, signaling, and accounting. In Figure 1-177, 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 1-177 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 1-177 Applying route-policies

In Figure 1-177, 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:

1. 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.
2. Assign different community attributes to the routes advertised by the PE in plane A and the routes advertised by the PE in plane B.
3. 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 1-60 IP addresses of physical interfaces

Local Device	Local Interface and Its IP Address	Remote Interface and Its IP Address	Remote Device
P1	GE 0/1/0 10.1.1.1/30	GE 0/1/0 10.1.1.2/30	P3
P1	GE 0/2/0 10.1.2.1/30	GE 0/1/0 10.1.2.2/30	P5
P1	GE 0/3/0 10.1.3.1/30	GE 0/1/0 10.1.3.2/30	RR
P1	GE 0/1/1 10.1.4.1/30	GE 0/1/0 10.1.4.2/30	P2

 Copyright © Huawei Technologies Co., Ltd.

Local Device	Local Interface and Its IP Address	Remote Interface and Its IP Address	Remote Device
P1	GE 0/1/2 10.1.5.1/30	GE 0/1/0 10.1.5.2/30	PE1
P2	GE 0/1/1 10.1.6.1/30	GE 0/1/0 10.1.6.2/30	P6
P2	GE 0/3/0 10.1.7.1/30	GE 0/1/0 10.1.7.2/30	P4
P2	GE 0/2/0 10.1.8.1/30	GE 0/2/0 10.1.8.2/30	RR
P2	GE 0/1/2 10.1.9.1/30	GE 0/1/0 10.1.9.2/30	PE2
P3	GE 0/2/0 10.1.10.1/30	GE 0/2/0 10.1.10.2/30	P5
P3	GE 0/3/0 10.1.11.1/30	GE 0/2/0 10.1.11.2/30	P4
P3	GE 0/1/1 10.1.12.1/30	GE 0/1/0 10.1.12.2/30	PE3
P4	GE 0/3/0 10.1.13.1/30	GE 0/3/0 10.1.13.2/30	P6
P4	GE 0/1/1 10.1.14.1/30	GE 0/1/0 10.1.14.2/30	PE4
P5	GE 0/3/0 10.1.15.1/30	GE 0/2/0 10.1.15.2/30	P6
PE1	GE 0/2/0 10.1.16.1/30	GE 0/2/0 10.1.16.2/30	PE2
PE3	GE 0/2/0 10.1.17.1/30	GE 0/2/0 10.1.17.2/30	PE4

Table 1-61 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

 Copyright © Huawei Technologies Co., Ltd.

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 1-62 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 File** 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 File** 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 vpnv4
[*PE1-bgp-af-vpnv4] peer 10.11.11.9 enable
[*PE1-bgp-af-vpnv4] 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 vpnv4
[*RR-bgp-af-vpnv4] undo policy vpn-target
[*RR-bgp-af-vpnv4] peer client enable
[*RR-bgp-af-vpnv4] peer 10.7.7.9 group client
[*RR-bgp-af-vpnv4] peer 10.8.8.9 group client
[*RR-bgp-af-vpnv4] peer 10.9.9.9 group client
[*RR-bgp-af-vpnv4] peer 10.10.10.9 group client
[*RR-bgp-af-vpnv4] peer client reflect-client
[*RR-bgp-af-vpnv4] commit
[~RR-bgp-af-vpnv4] 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 vpnv4 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 vpnv4 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 File** 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 File** 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  GigabitEthernet0/1/0
```

----End

Configuration File

- 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 GigabitEthernet0/1/0
 description toP3GE0/1/0
 undo shutdown
 ip address 10.1.1.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toP5GE0/1/0
 undo shutdown
 ip address 10.1.2.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 description toRRGE0/1/0
 undo shutdown
 ip address 10.1.3.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/1/1
 description toP2GE0/1/0
 undo shutdown
 ip address 10.1.4.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/1/2
 description toP2GE0/1/0
 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 GigabitEthernet0/1/0
 description toP1GE0/1/1
 undo shutdown
 ip address 10.1.4.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toRRGE0/2/0
 undo shutdown
 ip address 10.1.8.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 description toP4GE0/1/0
 undo shutdown
 ip address 10.1.7.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/1/1
 description toP6GE0/1/0
 undo shutdown
 ip address 10.1.6.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/1/2
 description toPE2GE0/1/0
 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 GigabitEthernet0/1/0
 description toP1GE0/1/0
 undo shutdown
 ip address 10.1.1.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toP5GE0/2/0
 undo shutdown
 ip address 10.1.10.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 description toP4GE0/2/0
 undo shutdown
 ip address 10.1.11.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/1/1
 description toPE3GE0/1/0
 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 GigabitEthernet0/1/0
 description toP2GE0/3/0
 undo shutdown
 ip address 10.1.7.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toP3GE0/3/0
 undo shutdown
```

```
 ip address 10.1.11.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 description toP6GE0/3/0
 undo shutdown
 ip address 10.1.13.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/1/1
 description toPE4GE0/1/0
 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 GigabitEthernet0/1/0
 description toP1GE0/2/0
 undo shutdown
 ip address 10.1.2.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toP3GE0/2/0
 undo shutdown
 ip address 10.1.10.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 description toP6GE0/2/0
 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
```

 Copyright © Huawei Technologies Co., Ltd.

- 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 GigabitEthernet0/1/0
 description toP2GE0/1/1
 undo shutdown
 ip address 10.1.6.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toP5GE0/3/0
 undo shutdown
 ip address 10.1.15.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 description toP4GE0/3/0
 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 GigabitEthernet0/1/0
 description toP1GE0/1/2
 undo shutdown
 ip address 10.1.5.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toPE2GE0/2/0
 undo shutdown
 ip address 10.1.16.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 undo shutdown
#
interface GigabitEthernet0/3/0.10
 ip binding vpn-instance NGN_Media
 vlan-type dot1q 10
 ip address 10.21.1.73 255.255.255.252
#
interface GigabitEthernet0/3/0.11
 ip binding vpn-instance NGN_Signaling
 vlan-type dot1q 11
 ip address 10.21.1.77 255.255.255.252
#
interface GigabitEthernet0/3/0.12
 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 vpnv4
  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
#
route-policy local_pre permit node 10
 if-match community-filter 1
 apply local-preference 200
#
ip community-filter 1 permit 65000:100
#
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 GigabitEthernet0/1/0
 description toP2GE0/1/2
 undo shutdown
 ip address 10.1.9.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toPE1GE0/2/0
 undo shutdown
 ip address 10.1.16.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 undo shutdown
#
```

```
interface GigabitEthernet0/3/0.10
 ip binding vpn-instance NGN_Media
 vlan-type dot1q 10
 ip address 10.21.1.13 255.255.255.252
#
interface GigabitEthernet0/3/0.11
 ip binding vpn-instance NGN_Signaling
 vlan-type dot1q 11
 ip address 10.21.1.17 255.255.255.252
#
interface GigabitEthernet0/3/0.12
 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 vpnv4
  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
#
route-policy local_pre permit node 10
 if-match community-filter 1
 apply local-preference 200
#
ip community-filter 1 permit 65000:200
#
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 GigabitEthernet0/1/0
 description toP3GE0/1/1
 undo shutdown
 ip address 10.1.12.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toPE4GE0/2/0
 undo shutdown
 ip address 10.1.17.1 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 undo shutdown
#
interface GigabitEthernet0/3/0.10
 ip binding vpn-instance NGN_Media
 vlan-type dot1q 10
 ip address 10.22.1.73 255.255.255.252
#
interface GigabitEthernet0/3/0.11
 ip binding vpn-instance NGN_Signaling
 vlan-type dot1q 11
 ip address 10.22.1.77 255.255.255.252
#
interface GigabitEthernet0/3/0.12
 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 vpnv4
```

```
 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
#
route-policy local_pre permit node 10
 if-match community-filter 1
 apply local-preference 200
#
route-policy local_pre permit node 20
#
ip community-filter 1 permit 65000:100
#
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 GigabitEthernet0/1/0
 description toP4GE0/1/1
```

```
 undo shutdown
 ip address 10.1.14.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toPE3GE0/2/0
 undo shutdown
 ip address 10.1.17.2 255.255.255.252
 mpls
 mpls ldp
 isis enable 64
#
interface GigabitEthernet0/3/0
 undo shutdown
#
interface GigabitEthernet0/3/0.10
 ip binding vpn-instance NGN_Media
 vlan-type dot1q 10
 ip address 10.22.1.13 255.255.255.252
#
interface GigabitEthernet0/3/0.11
 ip binding vpn-instance NGN_Signaling
 vlan-type dot1q 11
 ip address 10.22.1.17 255.255.255.252
#
interface GigabitEthernet0/3/0.12
 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 vpnv4
  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
#
route-policy local_pre permit node 10
 if-match community-filter 1
 apply local-preference 200
```

```
#
ip community-filter 1 permit 65000:200
#
return
```

- RR configuration file

```
#
sysname RR
#
isis 64
 network-entity 49.0091.0100.1101.1009.00
#
interface GigabitEthernet0/1/0
 description toP1GE0/3/0
 undo shutdown
 ip address 10.1.3.2 255.255.255.252
 isis enable 64
#
interface GigabitEthernet0/2/0
 description toP2GE0/2/0
 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
```

1.1.12.14.4 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 1-178, 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 1-178 Networking for configuring a basic or advanced VPN-target extended community filter

NOTE

Interface 1 in this example represents GE 0/1/0.

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 0/1/0
[~PE1-GigabitEthernet0/1/0] mpls
[*PE1-GigabitEthernet0/1/0] mpls ldp
[*PE1-GigabitEthernet0/1/0] commit
[~PE1-GigabitEthernet0/1/0] 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 0/1/0
[~PE2-GigabitEthernet0/1/0] mpls
[*PE2-GigabitEthernet0/1/0] mpls ldp
[*PE2-GigabitEthernet0/1/0] commit
[~PE2-GigabitEthernet0/1/0] 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 vpnb
[*PE1-vpn-instance-vpnb] ipv4-family
[*PE1-vpn-instance-vpnb-af-ipv4] route-distinguisher 2:100
[*PE1-vpn-instance-vpnb-af-ipv4] vpn-target 2:100 both
[*PE1-vpn-instance-vpnb-af-ipv4] quit
[*PE1-vpn-instance-vpnb] 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
```

 Copyright © Huawei Technologies Co., Ltd.

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 vpnv4 all routing-table** command on PE2 to check information about BGP VPNv4 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 vpnv4 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: GigabitEthernet0/1/0
 Relay Tunnel Out-Interface: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/0
 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 vpnv4 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: GigabitEthernet0/1/0
Relay Tunnel Out-Interface: GigabitEthernet0/1/0
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: GigabitEthernet0/1/0
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 vpnv4 all routing-table** command on PE2 to check information about BGP VPNv4 routes. The command output shows that route 4.4.4.4/32 has been filtered out.

```
[~PE2] display bgp vpnv4 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: GigabitEthernet0/1/0
 Relay Tunnel Out-Interface: GigabitEthernet0/1/0
 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: GigabitEthernet0/1/0
 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 vpnv4 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 GigabitEthernet0/1/0
```

```
 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 vpnv4
  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 GigabitEthernet0/1/0 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 GigabitEthernet0/1/0
 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 vpnv4
  policy vpn-target
  peer 1.1.1.1 enable
  peer 1.1.1.1 route-policy test import
```

```
#
route-policy test permit node 10
 if-match extcommunity-filter 1
#
route-policy test deny node 20
#
ip extcommunity-filter 1 index 10 permit rt 1:100
#
ip route-static 1.1.1.1 255.255.255.255 GigabitEthernet0/1/0 10.1.1.1
#
return
```

1.1.13 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.

1.1.13.1 Overview of XPL

This section describes the concept of XPL, the paragraph editing interface, set and route-filter configuration steps, and XPL clauses.

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, but it uses different editing and filtering methods from routing policies. Therefore, XPL can meet different customer requirements.

Table 1-63 compares XPL and routing policies.

Table 1-63 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.	Users can configure or modify policies as required in a text editor.
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.

NOTE

For details about routing policies, see "Routing Policies" in *HUAWEI NE40E-M2 seriesUniversal Service Router Feature Description — IP Routing*.

Line-by-Line and Paragraph-by-Paragraph Editing

XPL supports line-by-line editing and paragraph-by-paragraph editing, whereas routing policies support line-by-line editing only. Line-by-line editing is a traditional configuration method, whereas paragraph-by-paragraph editing is an innovative configuration method. **Table 1-64** compares the two methods.

Table 1-64 Line-by-line and paragraph-by-paragraph editing comparison

Item	Applicable to	Differences	Help and Error Correction Mechanisms
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 through line-by-line editing, enter the specific command view and reconfigure the set or policy.	The desired command can be suggested using the command association 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 command association function is not supported, and complete clauses must be entered in the paragraph editing UI. If any configuration error occurs, it is reported after the configurations of the whole paragraph are committed.

Paragraph Editing Interface

This section does not detail the line-by-line editing. For details about it, see the set or route-filter configuration process. 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.

Paragraph-by-paragraph editing is described in terms of **Paragraph Editing Interface** and **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL paragraph editing clauses, see **1.1.13.8 XPL Paragraph Editing Clauses**.

The paragraph editing interface functions as a text editor, in which users can edit XPL clauses to configure or modify sets and route-filters.

Common shortcut keys

Shortcut keys can be used in the paragraph editing interface. **Table 1-65** lists common shortcut keys and their functions.

Table 1-65 Shortcut keys commonly used in the 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.
Esc	Exits the text editing mode.
:q!	Exits the paragraph editing interface view without saving configurations. **NOTE** The shortcut key takes effect only after **Enter** is pressed.
:wq	Saves configurations and exits from the paragraph editing interface view. **NOTE** The shortcut key takes effect only after **Enter** is pressed.

Paragraph editing interface views

The following lists the different paragraph editing interface 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.

Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode

Sets and route-filters can be configured as follows:

- To configure a global variable set:
  a. Configure a start clause (**xpl global-value**) for a global variable set.
  b. 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.

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 routes that match the rules specified in the XPL 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.

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.

1.1.13.2 Configuration Precautions for XPL

Feature Requirements

Table 1-66 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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B
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.	NE40E-M2	NE40E-M2E/ NE40E-M2F/ NE40E-M2H/ NE40E-M2K/ NE40E-M2K-B

1.1.13.3 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 editing mode.
  a. Run the **edit xpl global-value** command to enter the global variable 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 global-value**) for a global variable set.
d. Configure set elements in the format of variable name+'value', for example, aaa '12', bbb'34', ccc '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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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

NOTE

If only 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 editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL clauses, see **1.1.13.8 XPL Paragraph Editing Clauses**.

Objective	Configure a global variable set that includes two values: 100 and ip-prefix.

Configuration Example	<HUAWEI> **edit xpl global-value** xpl global-value aaa '100', bbb 'ip-prefix', end-global-value The global variable set includes a set named **aaa** with value 100 and another set named **bbb** with value ip-prefix.
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	<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
	
	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**.

1.1.13.4 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 NE40E 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:

1.1.13.4.1 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 editing mode.
  a. Run the **edit xpl ip-prefix-list** *ip-prefix-list-name* command to enter the IPv4 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 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL clauses, see **1.1.13.8 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	<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 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.

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	<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 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.
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	<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
	
	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.

 Copyright © Huawei Technologies Co., Ltd.

1.1.13.4.2 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

     To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL clauses, see **1.1.13.8 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	<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 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.
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	<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 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.

 Copyright © Huawei Technologies Co., Ltd.

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	<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
	
	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.

1.1.13.4.3 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.
  a. Run the **edit xpl as-path-list** *as-path-list-name* command to enter the AS_Path 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 as-path-list** *as-path-list-name*) for an AS_Path set.
  d. 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.

e. 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL clauses, see **1.1.13.8 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	<HUAWEI> **edit xpl as-path-list aaa** xpl as-path-list aaa regular ^100_, regular _200_, regular _300$ end-list 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.
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	<HUAWEI> **edit xpl route-filter r1**
	xpl route-filter r1
	if as-path in aaa then
	refuse
	else
	approve
	endif
	end-filter
	
	The route-filter references AS_Path set **aaa**, denies the routes that match **aaa**, and permits all other routes.

1.1.13.4.4 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL clauses, see **1.1.13.8 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	<HUAWEI> **edit xpl community-list aaa** xpl community-list aaa 100:1, 200:1, 300:1, no-export end-list This community set includes four elements and matches the routes carrying community set 100:1, 200:1, 300:1, or no-export.
Objective	Configure a community set to match the routes carrying community set 100:*, 200:*, or 300:1.

Configuration Example 2	<HUAWEI> **edit xpl community-list bbb** xpl community-list bbb regular 100:*, regular 200:*, 300:1 end-list This community set includes three elements and matches the routes carrying community set 100:*, 200:*, or 300:1.
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	<HUAWEI> **edit xpl route-filter r1**
	
	xpl route-filter r1
	if local-preference le 100 then
	apply community aaa additive
	endif
	end-filter
	
	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.

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	<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
	
	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.

1.1.13.4.5 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.

     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 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

     To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For details about how to configure sets and route-filters, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For details about XPL clauses, see **1.1.13.8 XPL Paragraph Editing Clauses**.

Objectives	Configure a Large-Community set to match Large-Community attribute values 100:1:1, 200:1:1, and 300:1:1.
Configuration Example 1	<HUAWEI> **edit xpl large-community-list aaa** xpl large-community-list aaa 100:1:1, 200:1:1, 300:1:1 end-list This Large-Community set includes three elements and matches Large-Community attribute values 100:1:1, 200:1:1, and 300:1:1.
Objectives	Configure a Large-Community set to match Large-Community attribute values 100:1:*, 200:1:*, and 300:1:1.

Configuration Example 2	<HUAWEI> **edit xpl large-community-list bbb** xpl large-community-list bbb regular 100:1:*, regular 200:1:*, 300:1:1 end-list This Large-Community set includes three elements and matches Large-Community attribute values 100:1:*, 200:1:*, and 300:1:1.
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	<HUAWEI> **edit xpl route-filter r1**
	
	xpl route-filter r1
	if local-preference le 100 then
	apply large-community aaa additive
	endif
	end-filter
	
	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.

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	<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
	
	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.

1.1.13.4.6 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For detailed description about XPL clauses, see **1.1.13.8 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	<HUAWEI> **edit xpl rd-list rd-list1** xpl rd-list rd-list1 100:1, 200:1, regular 300:* end-list The RD set contains three elements and can match VPN routes carrying RD 100:1, 200:1, or 300:*.
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	<HUAWEI> **edit xpl route-filter r1**
	
	xpl route-filter r1
	if rd in rd-list1 then
	approve
	else
	refuse
	endif
	end-filter
	
	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.

1.1.13.4.7 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 0.0:0.
- **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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For detailed description about XPL clauses, see **1.1.13.8 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	<HUAWEI> **edit xpl extcommunity-list rt rt-list1** xpl extcommunity-list rt rt-list1 100:1, 200:1, 300:1 end-list The route target set contains three elements and can match VPN routes carrying RT 100:1, 200:1, or 300:1.
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	<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
	
	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.

1.1.13.4.8 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.

     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 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Configuration Procedures of Sets and Route-Filters Using the Paragraph Editing Mode**. For detailed description about XPL clauses, see **1.1.13.8 XPL Paragraph Editing Clauses**.

Objective	Configure an SoO set to match the VPN routes carrying the SoO (100:1, 200:1, or 300:1).
Configuration Example	<HUAWEI> **edit xpl extcommunity-list soo soo-list1** xpl extcommunity-list soo soo-list1 100:1, 200:1, 300:1 end-list The SoO set contains three elements and can match VPN routes carrying SoO 100:1, 200:1, or 300:1.
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	<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
	
	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.

1.1.13.4.9 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

1.1.13.5 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 editing interface 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 editing interface view. To commit the configurations, press **Y**.

To exit from the global variable set paragraph editing interface 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 editing is used as an example, the corresponding line editing is similar. To use the line editing mode, perform the operations described in paragraph editing.

For detailed set and route-filter configuration steps, see **Set and Route-Filter Configuration Steps**. For details about XPL clauses, see **1.1.13.8 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	<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
	
	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.

Objective	Create a route-filter with a pre-defined variable and configure another route-filter to reference the preceding route-filter.

Configuration Example 2	<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
	
	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.

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	<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
	
	The conditions in a condition clause can be connected with the Boolean operator **NOT**, **AND**, or **OR**.

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.

1.1.13.6 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 1-67** lists usage scenarios and relevant configuration tasks.

  Table 1-67 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 1-68** lists usage scenarios and relevant configuration tasks.

  Table 1-68 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**

 Copyright © Huawei Technologies Co., Ltd.

- Apply route-filters to RIPng.

  Route-filters can be used in interaction between RIPng and other routing protocols. **Table 1-69** lists usage scenarios and relevant configuration tasks.

  Table 1-69 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 1-70** lists usage scenarios and relevant configuration tasks.

  Table 1-70 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 1-71** lists usage scenarios and relevant configuration tasks.

  Table 1-71 Route-filter application in BGP

Usage Scenario	Relevant Configuration Task
Use a route-filter to set a priority for BGP routes.	**1.1.10.5.1 Setting the BGP Preference**
Use a route-filter to filter the routes to be imported by BGP.	**1.1.10.3.3 Configuring BGP to Import Routes**

 Copyright © Huawei Technologies Co., Ltd.

Usage Scenario	Relevant Configuration Task
Use a route-filter to filter the routes to be advertised by BGP.	**1.1.10.7.1 Using XPL to Filter the BGP Routes to Be Advertised**
Use a route-filter to filter the routes to be received by BGP.	**1.1.10.7.2 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 1-72** lists usage scenarios and relevant configuration tasks.

  Table 1-72 Route-filter application in BGP4+

Usage Scenario	Relevant Configuration Task
Use a route-filter to set a priority for BGP4+ routes.	**1.1.11.5.1 Configuring BGP4+ to Import Routes**
Use a route-filter to filter the routes to be imported by BGP4+.	**1.1.11.7.1 Setting a BGP4+ Preference**
Use a route-filter to filter the routes to be advertised by BGP4+.	**1.1.11.9.1 Using XPL to Filter the BGP4+ Routes to Be Advertised**
Use a route-filter to filter the routes to be received by BGP4+.	**1.1.11.9.2 Using XPL to Filter the BGP4+ Routes to Be Received**

----End

1.1.13.7 Configuration Examples for XPL

This section describes several XPL configuration examples.

1.1.13.7.1 Example for Using XPL to Filter the Routes to Be Received and Advertised (Paragraph-by-Paragraph 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 paragraph editing mode.

Networking Requirements

On the BGP network shown in **Figure 1-179**, Device A and Device B belong to AS 100, and Device C, Device D, and Device E belong to AS 200. Device A imports routes 2.2.2.0/24, 3.3.3.0/24, 4.4.4.0/24, and 5.5.5.0/24. The requirements are as follows:

- Device A advertises routes 2.2.2.0/24, 3.3.3.0/24, and 4.4.4.0/24 to Device B.
- On receipt of the three routes from Device B, Device C advertises all of them to Device E. On receipt of the three routes from Device B, Device D increases the MED value of the route 2.2.2.0/24 and advertises routes 2.2.2.0/24 and 3.3.3.0/24 to Device E. The MED of the route 2.2.2.0/24 learned from Device C is less than that of the route 2.2.2.0/24 learned from Device D. Therefore, Device E selects Device C 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 Device A, and two import policies are configured on Device E.

Figure 1-179 Cyclic figure for filtering the routes to be received and advertised

Device Name	Interface	IP Address
Device A	GE 0/3/0	1.1.5.2/24
Device B	GE 0/3/0	1.1.5.1/24
Device B	GE 0/3/1	1.1.4.2/24
Device B	GE 0/3/2	1.1.3.2/24
Device C	GE 0/3/1	1.1.1.2/24
Device C	GE 0/3/2	1.1.3.1/24
Device D	GE 0/3/1	1.1.2.2/24
Device D	GE 0/3/2	1.1.4.1/24
Device E	GE 0/3/1	1.1.2.1/24
Device E	GE 0/3/2	1.1.1.1/24

Precautions

When using XPL to filter the routes to be received and advertised, note the following points:

- A prefix range needs to be specified for the IPv4 prefix set as required.

- IPv4 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] quit
<DeviceA> edit xpl ip-prefix-list prefix1
xpl ip-prefix-list prefix1
2.2.2.0 24,
3.3.3.0 24,
4.4.4.0 24
end-list
```

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> edit xpl route-filter r1
xpl route-filter r1
if ip route-destination in prefix1 then
approve
else
refuse
endif
end-filter
```

NOTE

Press **i** in the paragraph editing interface view to enter the text editing mode.

After performing configurations in the text editing mode of the paragraph editing interface view, press **Esc** to exit from the text editing mode, and press **:wq** and **Enter** to save the configurations and exit from the paragraph editing interface view.

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] quit
<DeviceE> edit xpl route-filter appmed
xpl route-filter appmed($med)
```

```
apply med $med
end-filter
```

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> edit xpl route-filter r2
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 the route 2.2.2.0/24 to 100.

```
<DeviceE> edit xpl route-filter r3
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 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

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/3/0
 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
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 1.1.5.1 255.255.255.0
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.4.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

 Copyright © Huawei Technologies Co., Ltd.

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.2.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

- Device E configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.2.1 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

1.1.13.7.2 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 1-180**, Device A and Device B belong to AS 100, and Device C, Device D, and Device E belong to AS 200. Device A imports routes 2.2.2.0/24, 3.3.3.0/24, 4.4.4.0/24, and 5.5.5.0/24. The requirements are as follows:

- Device A advertises routes 2.2.2.0/24, 3.3.3.0/24, and 4.4.4.0/24 to Device B.
- On receipt of the three routes from Device B, Device C advertises all of them to Device E. On receipt of the three routes from Device B, Device D increases the MED value of the route 2.2.2.0/24 and advertises routes 2.2.2.0/24 and 3.3.3.0/24 to Device E. The MED of the route 2.2.2.0/24 learned from Device C is less than that of the route 2.2.2.0/24 learned from Device D. Therefore, Device E selects Device C 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 Device A, and two import policies are configured on Device E.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-180 Cyclic figure for filtering the routes to be received and advertised

Device Name	Interface	IP Address
Device A	GE 0/3/0	1.1.5.2/24
Device B	GE 0/3/0	1.1.5.1/24
Device B	GE 0/3/1	1.1.4.2/24
Device B	GE 0/3/2	1.1.3.2/24
Device C	GE 0/3/1	1.1.1.2/24
Device C	GE 0/3/2	1.1.3.1/24
Device D	GE 0/3/1	1.1.2.2/24
Device D	GE 0/3/2	1.1.4.1/24
Device E	GE 0/3/1	1.1.2.1/24
Device E	GE 0/3/2	1.1.1.1/24

Precautions

When using XPL to filter the routes to be received and advertised, note the following points:

- A prefix range needs to be specified for the IPv4 prefix set as required.
- IPv4 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
```

 Copyright © Huawei Technologies Co., Ltd.

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

- Device A configuration file

```
#
sysname DeviceA
#
interface GigabitEthernet0/3/0
 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
```

- Device B configuration file

```
#
sysname DeviceB
#
interface GigabitEthernet0/3/0
 undo shutdown
 ip address 1.1.5.1 255.255.255.0
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.4.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

- Device C configuration file

```
#
sysname DeviceC
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.1.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

- Device D configuration file

```
#
sysname DeviceD
#
interface GigabitEthernet0/3/1
```

```
 undo shutdown
 ip address 1.1.2.2 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

- Device E configuration file

```
#
sysname DeviceE
#
interface GigabitEthernet0/3/1
 undo shutdown
 ip address 1.1.2.1 255.255.255.0
#
interface GigabitEthernet0/3/2
 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
```

 Copyright © Huawei Technologies Co., Ltd.

1.1.13.8 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:

1.1.13.8.1 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 *ip-prefix-list-name*: begins the edition of an IPv4 prefix set.	**end-list**: concludes the edition of an IPv4 prefix set.
xpl ipv6-prefix-list *ipv6-prefix-list-name*: begins the edition of an IPv6 prefix set.	**end-list**: concludes the edition of an IPv6 prefix set.
xpl as-path-list *as-path-list-name*: begins the edition of an AS_Path set.	**end-list**: concludes the edition of an AS_Path set.

Configuration Start Clauses	Configuration End Clauses
xpl community-list *community-list-name*: begins the edition of a community set.	**end-list**: concludes the edition of a community set.
xpl large-community-list *large-community-list-name*: starts the edition of a Large-community set.	**end-list**: concludes the edition of a Large-community set.
xpl rd-list *rd-list-name*: begins the edition of an RD set.	**end-list**: concludes the edition of an RD set.
xpl extcommunity-list rt *rt-list-name*: begins the edition of a route target set.	**end-list**: concludes the edition of a route target set.
xpl extcommunity-list soo *soo-list-name*: begins the edition of a site of origin (SoO) set.	**end-list**: concludes the edition of an SoO set.
xpl route-filter *route-filter-name*: begins the edition of a route-filter.	**end-filter**: concludes the edition of a route-filter.
xpl route-flow-group *group-name*: 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**.

1.1.13.8.2 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...}.

 Copyright © Huawei Technologies Co., Ltd.

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 *med*.	*med*: indicates an integer ranging from 0 to 4294967295.	• **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** *vpn-instance-name* **protocol ospf** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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 *med*.	*med*: indicates an integer ranging from 0 to 4294967295.	• **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** *vpn-instance-name* **protocol ospf** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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 *med*.	*med*: indicates an integer ranging from 0 to 4294967295.	• **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** *vpn-instance-name* **protocol ospf** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
{ **isis-cost** | **rip-cost** | **ospf-cost** } **eq** *cost*	Matches the IS-IS/RIP/OSPF routes with the specified *cost*.	*cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (OSPF)** • **default-route-advertise (OSPF)** • **preference (OSPF)** • **preference (OSPFv3)** • **local-mt filter-policy (OSPF)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **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** *vpn-instance-name* **protocol isis** • **preference (RIP)** • **import-route (RIP)** • **default-route originate (RIP)** • **preference (RIPng)** • **import-route (RIPng)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
{ **isis-cost** | **rip-cost** | **ospf-cost** } **ge** *cost*	Matches the IS-IS/RIP/OSPF routes with the cost greater than or equal to the specified *cost*.	*cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (OSPF)** • **default-route-advertise (OSPF)** • **preference (OSPF)** • **preference (OSPFv3)** • **local-mt filter-policy (OSPF)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **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** *vpn-instance-name* **protocol isis** • **preference (RIP)** • **import-route (RIP)** • **default-route originate (RIP)** • **preference (RIPng)** • **import-route (RIPng)**

 Copyright © Huawei Technologies Co., Ltd.

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 *cost*.	*cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (OSPF)** • **default-route-advertise (OSPF)** • **preference (OSPF)** • **preference (OSPFv3)** • **local-mt filter-policy (OSPF)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **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** *vpn-instance-name* **protocol isis** • **preference (RIP)** • **import-route (RIP)** • **default-route originate (RIP)** • **preference (RIPng)** • **import-route (RIPng)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
rib-cost eq *cost*	Matches all routes with the specified *cost* in the IP routing table.	*cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (BGP)** • **network (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol isis** • **import-route (OSPF)** • **default-route-advertise (OSPF)** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis**
rib-cost ge *cost*	Matches all routes with the cost greater than or equal to the specified *cost* in the IP routing table.	*cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (BGP)** • **network (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol isis** • **import-route (OSPF)** • **default-route-advertise (OSPF)** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis**

 Copyright © Huawei Technologies Co., Ltd.

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 *cost* in the IP routing table.	*cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (BGP)** • **network (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol isis** • **import-route (OSPF)** • **default-route-advertise (OSPF)** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
preference eq *preference*	Matches the routes with the specified *preference*.	*preference*: indicates an integer ranging from 0 to 255.	• **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** *vpn-instance-name* **protocol 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)** • **preference (RIP)** • **preference (RIPng)** • **import-route (RIP)** • **import-route (RIPng)** • **default-route originate (RIP)** • **ripng default-route**

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
preference le *preference*	Matches the routes whose priority is less than or equal to the specified *preference*.	*preference*: indicates an integer ranging from 0 to 255.	• **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** *vpn-instance-name* **protocol 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)** • **preference (RIP)** • **preference (RIPng)** • **import-route (RIP)** • **import-route (RIPng)** • **default-route originate (RIP)** • **ripng default-route**

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
preference ge *preference*	Matches the routes whose priority is greater than or equal to the specified *preference*.	*preference*: indicates an integer ranging from 0 to 255.	• **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** *vpn-instance-name* **protocol 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)** • **preference (RIP)** • **preference (RIPng)** • **import-route (RIP)** • **import-route (RIPng)** • **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 IPv4 routes carrying the specified source IP address.	*ip-prefix-list-name*: 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 (:). *ip-prefix-list*: 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 }.	• **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)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			• **import-route (OSPF)** • **import-route (OSPFv3)** • **preference (OSPF)** • **preference (OSPFv3)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol 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** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
ipv6 route-source in { *ipv6-prefix-list-name* | *ipv6-prefix-list* }	Matches the IPv6 routes carrying the specified source IPv6 address.	*ipv6-prefix-list-name*: 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 (:). *ipv6-prefix-list*: indicates the IPv6 prefix set in the format of {element A, element B...}, in which the elements are IPv6 addresses carrying mask lengths.	• **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** *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**

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			• **ip import-rib vpn-instance** *vpn-instance-name* **protocol 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** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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.	*ip-prefix-list-name*: 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 (:). *ip-prefix-list*: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths.	• **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** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **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** • **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
			• **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** *vpn-instance-name* **protocol 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** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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.	*ipv6-prefix-list-name*: 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 (:). *ipv6-prefix-list*: indicates the IPv6 prefix set in the format of {element A, element B...}, in which the elements are IPv6 addresses carrying mask lengths.	• **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** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **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** • **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
			• **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** *vpn-instance-name* **protocol 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** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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.	*ip-prefix-list-name*: 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 (:). *ip-prefix-list*: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths.	• **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** *vpn-instance-name* } **protocol static** • **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** • **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
			• **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** *vpn-instance-name* **protocol ospf** • **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** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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.	*ipv6-prefix-list-name*: 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 (:). *ipv6-prefix-list*: indicates the IPv6 prefix set in the format of {element A, element B...}, in which the elements are IPv6 addresses carrying mask lengths.	• **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** *vpn-instance-name* } **protocol static** • **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** • **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
			• **ip import-rib vpn-instance** *vpn-instance-name* **protocol 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** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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 *tag*.	*tag*: indicates an integer ranging from 0 to 4294967295.	• **import-route (BGP)** • **network (BGP)** • **peer route-filter import** • **peer route-filter export** • **preference (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **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** • **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** *vpn-instance-name* **protocol ospf** • **filter-policy import (OSPF)** • **filter-policy import (OSPFv3)** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **preference (IS-IS)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			• **frr-policy route (IS-IS)** • **filter-policy export (IS-IS)** • **filter-policy import (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

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*.	*tag*: indicates an integer ranging from 0 to 4294967295.	• **import-route (BGP)** • **network (BGP)** • **peer route-filter import** • **peer route-filter export** • **preference (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **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** • **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** *vpn-instance-name* **protocol ospf** • **filter-policy import (OSPF)** • **filter-policy import (OSPFv3)** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **preference (IS-IS)** • **frr-policy route (IS-IS)** • **filter-policy export (IS-IS)**

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			• **filter-policy import (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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 *tag*.	*tag*: indicates an integer ranging from 0 to 4294967295.	• **import-route (BGP)** • **network (BGP)** • **peer route-filter import** • **peer route-filter export** • **preference (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **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** • **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** *vpn-instance-name* **protocol ospf** • **filter-policy import (OSPF)** • **filter-policy import (OSPFv3)** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **preference (IS-IS)** • **frr-policy route (IS-IS)** • **filter-policy export (IS-IS)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			• **filter-policy import (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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 *preference*	Matches the BGP routes with the Local_Pref value being *preference*.	*preference*: indicates an integer ranging from 0 to 4294967295.	• **peer route-filter import** • **peer route-filter export** (This command takes effect only for IBGP peers.)
local-preference ge *preference*	Matches the BGP routes with the Local_Pref value greater than or equal to *preference*.	*preference*: indicates an integer ranging from 0 to 4294967295.	• **peer route-filter import** • **peer route-filter export** (This command takes effect only for IBGP peers.)
local-preference le *preference*	Matches the BGP routes with the Local_Pref value less than or equal to *preference*.	*preference*: indicates an integer ranging from 0 to 4294967295.	• **peer route-filter import** • **peer route-filter export** (This command takes effect only for IBGP peers.)

 Copyright © Huawei Technologies Co., Ltd.

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.	*as-path-list-name*: 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 (:). *as-path-list*: 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.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

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).	–	• **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 *as-path* [**whole-match**]	Matches BGP routes with AS_Path whose rightmost AS numbers are the same as *as-path*.	*as-path*: 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.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

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 *as-path*.	*as-path*: 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.	• **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 *as-path*.	*as-path*: 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.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path length { **eq** | **ge** | **le** } *as-path-number*	Matches the BGP routes carrying the AS_Path with the length equal to *as-path-number*, greater than or equal to *as-path-number*, or less than or equal to *as-path-number*. • **eq**: equal to • **ge**: greater than or equal to • **le**: less than or equal to	*as-path-number*: indicates the AS_Path length. The value is an integer ranging from 0 to 2047.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
as-path unique-length { **eq** | **ge** | **le** } *as-path-number*	Matches the BGP routes carrying the AS_Path with the length equal to *as-path-number*, greater than or equal to *as-path-number*, or less than or equal to *as-path-number* (duplicate AS numbers are counted as one). • **eq**: equal to • **ge**: greater than or equal to • **le**: less than or equal to	*as-path-number*: indicates the AS_Path length. The value is an integer ranging from 0 to 2047.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

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.	*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 (:). *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**).	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
community matches-all { *community-list-name* | *community-list* }	Matches the BGP routes carrying the communities listed in the specified community set or more communities.	*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 (:). *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**).	• **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 { *community-list-name* | *community-list* }	Matches the BGP routes carrying the communities listed in the specified community set or less communities.	*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 (:). *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**).	• **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.	-	• **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.	*rd-list-name*: 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 (:). *rd-list*: indicates the RD set in the format of {element A, element B...}. For details about RD element values, see RD element values.	• **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.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. For details about RT element values, see RT element values.	• **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 { *rt-list-name* | *rt-list* }	Matches the VPN routes carrying the RTs listed in the specified route target set or more RTs.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. For details about RT element values, see RT element values.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

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.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. For details about RT element values, see RT element values.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity rt is-none	Matches the VPN routes with an empty RT.	-	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity soo matches-any { *soo-list-name* | *soo-list* }	Matches the VPN routes carrying at least one of the site or origins (SoOs) specified in the SoO set.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. For details about SoO element values, see SoO element values.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity soo matches-all { *soo-list-name* | *soo-list* }	Matches the VPN routes carrying the SoOs listed in the specified SoO set or more SoOs.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. For details about SoO element values, see SoO element values.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
extcommunity soo matches-within { *soo-list-name* | *soo-list* }	Matches the VPN routes carrying the SoOs listed in the specified SoO set or less SoOs.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. For details about SoO element values, see SoO element values.	• **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.	-	• **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)

 Copyright © Huawei Technologies Co., Ltd.

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.	–	• **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)

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
origin is *origin-type*	Matches the BGP routes carrying the specified *origin-type*.	The value of *origin-type* can be any of the following items: • **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.	• **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 *protocol-type*	Matches the routes of the specified routing protocol.	The value of *protocol-type* can be any of the following items: • **bgp** • **direct** • **isis** • **ospf** • **rip** • **static**	• **network (BGP)** • **peer route-filter import** • **peer route-filter export** • **aggregate suppress-filter** • **aggregate origin-filter** • **nexthop recursive-lookup (BGP)**
protocol in *protocol-type*	Matches the routes of the specified routing protocol.	The value of *protocol-type* can be any of the following items: • **bgp** • **direct** • **isis** • **ospf** • **rip** • **static**	• **network (BGP)** • **peer route-filter import** • **peer route-filter export** • **aggregate suppress-filter** • **aggregate origin-filter** • **nexthop recursive-lookup (BGP)**

 Copyright © Huawei Technologies Co., Ltd.

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
route-type is *route-type*	Matches the routes of the specified type.	The value of *route-type* can be any of the following items: • **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.	• **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** *vpn-instance-name* **protocol ospf** • **frr-policy route (OSPF)** • **frr-policy route (OSPFv3)** • **local-mt filter-policy (OSPF)** • **filter-policy import (OSPF)** • **filter-policy import (OSPFv3)** • **import-route (IS-IS)**

Condition Clause	Function	Parameters	Commands Used to Apply Condition Clauses
			• **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis** • **preference (IS-IS)** • **frr-policy route (IS-IS)** • **filter-policy export (IS-IS)** • **filter-policy import (IS-IS)**
path-type is *path-type*	Matches the BGP routes of the specified type.	The value of *path-type* can be either of the following items: • **ibgp**: indicates IBGP routes. • **ebgp**: indicates EBGP routes.	• **peer route-filter import** • **peer route-filter export**
rpki is *rpki-type*	Matches the BGP routes with the specified origin AS validation result.	The value of *rpki-type* can be any of the following items: • 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.	• **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)

 Copyright © Huawei Technologies Co., Ltd.

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.	*ip-prefix-list-name*: 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 (:). *ip-prefix-list*: indicates the IPv4 prefix set in the format of {element A, element B...}, in which the elements are IPv4 addresses carrying mask lengths.	None
interface in *interface-list-name*	Matches the BGP routes of which the outbound interfaces are in the specified outbound interface list.	*interface-list-name*: 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. *interface-list-name* must have been set using the **xpl interface-list** *section-name* command.	• **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**

1.1.13.8.3 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.	-

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses	Function	Parameters
call route-filter *route-filter-name*	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.	–

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses Used to Set Common Protocol Attributes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
ip next-hop { *ipv4-address* | **blackhole** | **peer-address** }	Sets a next hop IPv4 address.	*ipv4-address*: indicates an IPv4 address in dotted decimal notation. **blackhole**: adds a black-hole flag to a route. **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.	• **peer route-filter import** • **import-route (OSPF)** • **default-route-advertise (OSPF)** • **import-route (RIP)** • **default-route originate (RIP)** The *ipv4-address* and **peer-address** parameters are invalid in the following commands: • **import-route (BGP)** • **network (BGP)** The **blackhole** parameter is invalid in following commands: • **peer route-filter export** **NOTE** In the same route-filter, **ip next-hop** *ipv4-address* and **ip next-hop peer-address** commands cannot be both configured. Otherwise, the result may be unexpected.
ipv6 next-hop { *ipv6-address* | **blackhole** | **peer-address** }	Sets a next hop IPv6 address.	*ipv6-address*: indicates an IPv6 address. The value is a 32-digit hexadecimal number, in the format of X:X:X:X:X:X:X:X. **blackhole**: adds a black-hole flag to a route. **peer-address**: sets the next hop of a BGP route to the IPv4 address of a peer.	• **peer route-filter import** • **import-route (RIPng)** The *ipv6-address* and **peer-address** parameters are invalid in the following commands: • **import-route (BGP)** • **network (BGP)** • **import route-filter** The **blackhole** parameter is invalid in following commands: • **peer route-filter export** **NOTE** In the same route-filter, **ipv6 next-hop** *ipv6-address* and **ipv6 next-hop peer-address** commands cannot be both configured. Otherwise, the result may be unexpected.

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
preference *preference*	Sets a priority for a routing protocol. When multiple dynamic routing protocols run on a device and they discover multiple routes to the same destination, the priorities specified for the protocols can be used for route selection. The route discovered by the protocol with the highest priority (smallest priority value) is selected for IP packet forwarding.	*preference*: indicates the route priority. The value is an integer ranging from 1 to 255. The smaller the value, the higher the priority.	• **preference (BGP)** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol ospf** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol isis** • **preference (OSPF)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **preference (OSPFv3)** • **preference (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis** • **preference (RIP)** • **preference (RIPng)** • **default-route originate (RIP)**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
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.	• **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **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** *vpn-instance-name* **protocol isis**
aigp { [+ | -] *cost* | **inherit-cost** }	Sets the AIGP value.	+: increases the AIGP value of routes. -: reduces the AIGP value of routes. *cost*: indicates the AIGP value. **inherit-cost**: inherits the costs of the routes as AIGP values.	• **import-route (BGP)** • **network (BGP)** The **inherit-cost** parameter is invalid in following commands: • **peer route-filter import** • **peer route-filter export**
mpls-label	Enables MPLS to allocate labels to routes.	-	• **peer route-filter import** • **peer route-filter export**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
cost-type *cost-type*	Sets the cost type.	*cost-type* can be any of the following items: • **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	• **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** • **peer route-filter export** • **import-route (OSPF)** • **import-route (OSPFv3)** • **default-route-advertise (OSPF)** • **default-route-advertise (OSPFv3)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		of the IGP routes to which the BGP routes recurse. • **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.	
gateway-ip { **origin-nexthop** | *ip-address* }	Sets a gateway IP address for a route.	**origin-nexthop**: sets the gateway IPv4 address to the next-hop IPv4 address of the route. *ip-address*: sets the gateway IPv4 address to a specified IPv4 address.	• **peer route-filter import** • **peer route-filter export**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
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.	• **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
community { *community-list-name* | *community-list* } **overwrite**	Overwrites the original community attribute with the specified community set.	*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 (:). *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**).	• **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 address family view)

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
community { *community-list-name* | *community-list* } **additive**	Adds the specified community set to the original community attribute.	*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 (:). *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**).	• **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 address family view)
community in { *community-list-name* | *community-list* } **delete**	Deletes the communities specified in the community set from the original community attribute.	*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 (:). *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**).	• **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 address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
community not in { *community-list-name* | *community-list* } **delete**	Deletes the communities that are beyond the community set from the original community attribute.	*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 (:). *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**).	• **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 address family view)
community none	Deletes all the communities from the original community attribute.	-	• **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 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
as-path *as-path* **overwrite**	Sets the AS_Path attribute for BGP routes.	*as-path*: indicates the AS_Path, enclosed in single quotation marks, with every two neighboring AS numbers separated with a space.	• **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 address family view)
as-path { **as-number-plain** | **as-number-dot** } [*count*] **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 *x.y*, where *x* and *y* are integers ranging from 1 to 65535 and from 0 to 65535, respectively. *count*: indicates the number of times that an AS number is added to the AS_Path attribute of BGP routes.	• **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 address family view)

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
as-path *as-path* **substitute**	Replaces the AS number of the peer specified in the AS_Path attribute with the local AS number.	*as-path*: 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.	• **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 address family view)
as-path *as-path* **substitute whole-match**	Replaces the AS number of the peer specified in the AS_Path attribute with the local AS number in the case of whole-match.	*as-path*: 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.	• **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 address family view)

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses Used to Set Attributes Related to BGP Route Selection

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
med { [**+** | **-**] *med* | **igp-cost** }	Sets a value for the MED attribute of BGP routes.	**+**: increases the MED value. **-**: decreases the MED value. *med*: indicates an integer ranging from 0 to 4294967295. When **+** or **-** is specified, *med* indicates the value to be increased or reduced by. **igp-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.	• **peer route-filter export** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol ospf** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol isis** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol isis** The **igp-cost** parameter is invalid in following commands: • **import-route (BGP)** • **network (BGP)** • **peer route-filter import** • **aggregate attribute-filter** • **peer default-route-advertise** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol direct** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **import route-filter** • **export route-filter** • **import-rib** (BGP-VPN instance address family view) • **import-rib vpn-instance** (BGP public address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
local-preference [**+** | **-**] *local-preference*	Sets the Local_Pref attribute for BGP routes. The larger the value, the higher the priority.	*local-preference*: indicates an integer ranging from 0 to 4294967295. **+**: increases the Local_Pref value. **-**: decreases the Local_Pref value.	• **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 address family view)
preferred-value *preferred-value*	Sets the PrefVal attribute for BGP routes. During route selection, the route with the largest PrefVal is preferred.	*preferred-value*: indicates the PrefVal attribute. The value is an integer ranging from 0 to 65535.	• **import-route (BGP)** • **network (BGP)** • **peer route-filter import** • **aggregate attribute-filter** • **import-rib** (BGP-VPN instance address family view) • **import-rib vpn-instance** (BGP public address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
origin *origin-type*	Sets the origin type for BGP routes.	*origin-type* can be any of the following items: • **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.	• **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 address family view)

Action Clauses Used to Set BGP-Specific Attributes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
dampening *half-life reuse suppress max-suppress*	Sets dampening parameters for BGP routes.	*half-life*: indicates the half life of reachable routes. The value is an integer ranging from 1 to 45, in minutes. *reuse*: 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. *suppress*: 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 *reuse*. *max-suppress*: indicates the maximum penalty value. The value is an integer ranging from 1001 to 20000 and must be greater than *suppress*.	• **dampening**
group *group-name*	Sets a QPPB configuration group.	*group-name*: 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 (:).	• **peer route-filter import** • **import route-filter** • **qos-local-id** • **filter-policy import (OSPF)** • **local-mt filter-policy (OSPF)**

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses Used to Set IGP-Specific Attributes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
ospf-cost [**+** | **-** | **inherit**] *ospf-cost*	Sets a value for the cost of OSPF routes.	**+**: increases the cost value. **-**: decreases the cost value. **inherit-cost**: inherits the costs of the routes. *ospf-cost*: indicates an integer ranging from 0 to 4294967295.	• **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol ospf** • **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol ospfv3** • **default-route-advertise (OSPF)** • **default-route-advertise (OSPFv3)** • **import-route (OSPF)** • **import-route (OSPFv3)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **filter export (OSPF)** • **filter import (OSPF)** • **filter export (OSPFv3)** • **filter import (OSPFv3)**
isis-cost [**+** | **-** | **inherit**] *isis-cost*	Sets a value for the cost of IS-IS routes.	**+**: increases the cost value. **-**: decreases the cost value. **inherit-cost**: inherits the costs of the routes. *isis-cost*: indicates an integer ranging from 0 to 4294967295.	• **import-route (IS-IS)** • **default-route-advertise (IS-IS)**
rip-cost [**+** | **-**] *rip-cost*	Sets a value for the cost of RIP routes.	**+**: increases the cost value. **-**: decreases the cost value. *rip-cost*: indicates the cost of RIP routes. The value is an integer ranging from 0 to 16.	• **import-route (RIP)** • **import-route (RIPng)** • **default-route originate (RIP)**

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
isis *isis-level-type*	Sets the level of the routes imported by IS-IS.	*isis-level-type* can be any of the following items: • **level-1** • **level-1-2** • **level-2**	• **import-route (IS-IS)** • **default-route-advertise (IS-IS)**
ospf *ospf-sub-type*	Imports routes to a specified OSPF area.	*ospf-sub-type* can be either of the following items: • **backbone**: indicates the backbone area. • **stub-area**: indicates an NSSA.	None

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
tag *tag*	Sets a tag for IGP routes.	*tag*: indicates the tag of IGP routes. The value is an integer ranging from 0 to 4294967295.	• **import-rib** { **public** | **vpn-instance** *vpn-instance-name* } **protocol static** • **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** • **import-route (OSPF)** • **import-route (OSPFv3)** • **default-route-advertise (OSPF)** • **default-route-advertise (OSPFv3)** • **ip import-rib vpn-instance** *vpn-instance-name* **protocol ospf** • **import-route (IS-IS)** • **default-route-advertise (IS-IS)** • **ip import-rib vpn-instance** *vpn-instance-name* **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)**

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
rip-tag *rip-tag*	Sets a tag for RIP routes.	*rip-tag*: indicates the tag of RIP routes. The value is an integer ranging from 0 to 65535.	• **import-route (OSPF)** • **import-route (OSPFv3)** • **default-route-advertise (OSPF)** • **default-route-advertise (OSPFv3)** • **import-route (RIP)** • **import-route (RIPng)** • **default-route originate (RIP)**

 Copyright © Huawei Technologies Co., Ltd.

Action Clauses Used to Set the RT Attribute for VPN Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
extcommunity rt { *rt-list-name* | *rt-list* } **overwrite**	Overwrites the original RT with the specified RT.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. The elements can be expressed 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. • 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	• **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 vpn-instance** (BGP public address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		4294967295, and the user-defined number is an integer ranging 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. 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
extcommunity rt { *rt-list-name* | *rt-list* } **additive**	Adds the specified RT to the original RT.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. The elements can be expressed 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. • 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	• **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 vpn-instance** (BGP public address family view)

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		an integer ranging 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. 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
extcommunity rt in { *rt-list-name* | *rt-list* } **delete**	Deletes the RTs that are contained in the specified route target set.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. The elements can be expressed 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. ● 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
		an integer ranging 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. 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
extcommunity rt not in { *rt-list-name* | *rt-list* } **delete**	Deletes the RTs that are not contained in the specified route target set.	*rt-list-name*: 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 (:). *rt-list*: indicates the route target set in the format of {element A, element B...}. The elements can be expressed 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. ● 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
		an integer ranging 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. 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.	
extcommunity 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
extcommunity soo { *soo-list-name* | *soo-list* } **overwrite**	Overwrites the original SoO with the specified SoO.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed 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. ● 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	● **import-route (BGP)** ● **network (BGP)** ● **peer route-filter import** ● **peer route-filter export**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		4294967295, and the user-defined number is an integer ranging 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. 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
extcommunity soo { *soo-list-name* | *soo-list* } **additive**	Adds the specified SoO to the original SoO.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed 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. ● 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	● **import-route (BGP)** ● **network (BGP)** ● **peer route-filter import** ● **peer route-filter export**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		an integer ranging 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. 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
extcommunity soo in { *soo-list-name* | *soo-list* } **delete**	Deletes the SoOs that are contained in the specified SoO set.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed 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. ● 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	● **import-route (BGP)** ● **network (BGP)** ● **peer route-filter import** ● **peer route-filter export** **NOTE** The preceding commands can be run, but their configurations do not take effect.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		an integer ranging 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. 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
extcommunity soo not in { *soo-list-name* | *soo-list* } **delete**	Deletes the SoOs that are not contained in the specified SoO set.	*soo-list-name*: 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 (:). *soo-list*: indicates the SoO set in the format of {element A, element B...}. The elements can be expressed 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. ● 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	● **import-route (BGP)** ● **network (BGP)** ● **peer route-filter import** ● **peer route-filter export** **NOTE** The preceding commands can be run, but their configurations do not take effect.

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
		an integer ranging 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. 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.	
extcommunity soo none	Deletes all SoOs from VPN routes.	-	● **import-route (BGP)** ● **network (BGP)** ● **peer route-filter import** ● **peer route-filter export** **NOTE** The preceding commands can be run, but their configurations do not take effect.

Action Clauses Used to Set Redirection Extended Community Attributes for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
extcommunity redirect ip *ip-extcommunity-value*	Sets the IP redirection extended community attribute for BGP routes.	*ip-extcommunity-value*: indicates the IP redirection extended community attribute in the format of *ip-address:nn*, in which *ip-address* is in dotted decimal notation and *nn* is 0 or 1.	● **peer route-filter import** ● **peer route-filter export** ● **import-route (BGP)** ● **network (BGP)**

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
extcommunity redirect vpn-target *vpnrt-extcommunity-value*	Sets the VPN redirection extended community attribute for BGP routes.	*vpnrt-extcommunity-value*: indicates the VPN redirection extended community attribute in the format of *as-number:nn*, in which *as-number* is an AS number with the value being an integer ranging from 0 to 65535 and *nn* is an integer ranging from 0 to 4294967295.	• **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
extcommunity color *extcommunityvalue*	Sets a color extended community attribute for BGP routes matching specified filtering rules.	*extcommunityvalue*: 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.	• **peer route-filter import** • **peer route-filter export** • **import route-filter** • **export route-filter**

 Copyright © Huawei Technologies Co., Ltd.

Setting a Link Bandwidth Extended Community Attribute for BGP Routes

Action Clauses	Function	Parameters	Commands Used to Apply Action Clauses
extcommunity bandwidth *bandwidth-extcommunity-value* **overwrite**	Sets a link bandwidth extended community attribute for BGP routes.	*bandwidth-extcommunity-value*: 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.	• **peer route-filter import** • **peer route-filter export**
extcommunity bandwidth none	Deletes the link bandwidth extended community attribute set for BGP routes.	-	• **peer route-filter import** • **peer route-filter export**
extcommunity bandwidth aggregate limit *bandwidth-value*	Specifies the maximum value of the bandwidth extended community attribute.	*bandwidth-value*: specifies the link bandwidth. The value is an integer that ranges from 1 to 4294967295, in kbit/s.	• **peer route-filter export**
extcommunity bandwidth aggregate	Specifies the aggregated bandwidth extended community attribute for BGP routes.	-	• **peer route-filter export**

 Copyright © Huawei Technologies Co., Ltd.

1.1.14 Route Monitoring Group Configuration

1.1.14.1 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.

1.1.14.2 Configuration Precautions for Route Monitoring Group

Feature Requirements

None

1.1.14.3 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.

 Copyright © Huawei Technologies Co., Ltd.

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 1-181**, 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.

 Copyright © Huawei Technologies Co., Ltd.

Figure 1-181 Route monitoring group

On the network shown in **Figure 1-181**, 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

Step 1 Run **system-view**

The system view is displayed.

Step 2 Run **ip route-monitor-group** *group-name*

A route monitoring group is created, and the route monitoring group view is displayed.

Step 3 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.

Step 4 (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 link switchback.

In actual applications, 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. This process takes some time. If the RM module instructs the associated service module to perform a link switchover immediately, packet loss may occur. To address 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 switchover is performed after the corresponding forwarding entry is generated.

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 threshold of the service module.

Step 5 Run **monitor enable**

The route monitoring group is enabled.

When a large number of routes are being 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 disassociate the route monitoring group from all service modules. After configuration is complete, run the **monitor enable** command to re-associate the route monitoring group with the service modules.

Step 6 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.

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.

1.1.14.4 Configuration Examples for Route Monitoring Groups

1.1.14.4.1 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 1-182**, 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 1-182 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 0/1/0, GE 0/2/0, and GE0/3/0.100 respectively.

Device Name	Interface	IP Address
BRAS1	LoopBack1	1.1.1.1/32

Device Name	Interface	IP Address
	GE 0/3/0.100	10.10.10.1/24
	GE 0/2/0	10.1.2.1/24
BRAS2	LoopBack1	2.2.2.2/32
	GE 0/1/0	10.1.4.1/24
	GE 0/3/0.100	10.10.10.10/24
CR	LoopBack1	3.3.3.3/32
	GE 0/1/0	10.1.4.2/24
	GE 0/2/0	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 0/3/0.100, 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.

 Copyright © Huawei Technologies Co., Ltd.

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 0/3/0.100
[*BRAS1-Gigabitethernet0/3/0.100] vrrp vrid 3 virtual-ip 10.10.10.2
[*BRAS1-Gigabitethernet0/3/0.100] admin-vrrp vrid 3
[*BRAS1-Gigabitethernet0/3/0.100] vrrp vrid 3 priority 120
[*BRAS1-Gigabitethernet0/3/0.100] vrrp vrid 3 preempt-mode timer delay 1200
[*BRAS1-Gigabitethernet0/3/0.100] vrrp vrid 3 track bfd-session 2 peer
[*BRAS1-Gigabitethernet0/3/0.100] vrrp recover-delay 20
[*BRAS1-Gigabitethernet0/3/0.100] commit
[~BRAS1-Gigabitethernet0/3/0.100] 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 0/3/0.100
[*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 GigabitEthernet0/3/0.100
#
interface GigabitEthernet0/2/0
 ip address 10.1.2.1 255.255.255.0
#
interface GigabitEthernet0/3/0.100
 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 0.0.0.0
  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 GigabitEthernet0/3/0.100
#
interface GigabitEthernet0/1/0
 ip address 10.1.4.1 255.255.255.0
#
interface GigabitEthernet0/3/0.100
 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 0.0.0.0
  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 GigabitEthernet0/1/0
 ip address 10.1.4.2 255.255.255.0
#
interface GigabitEthernet0/2/0
 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 0.0.0.0
  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
```