JUNIPER NETWORKS | Engineering Simplicity

Network Configuration Example

Configuring an EVPN-VXLAN Fabric for a Campus Network With CRB

Published
2022-09-30

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Network Configuration Example Configuring an EVPN-VXLAN Fabric for a Campus Network With CRB

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About This Network Configuration Example | v

1 Example: How to Configure an EVPN-VXLAN Fabric for a Campus Network using CRB

Technology Primer: EVPN-VXLAN Fabrics for the Campus | 2

- Overview of EVPN-VXLAN-Based Campus Networks | 2
- EVPN-VXLAN Technical Overview | 4
- Campus EVPN-VXLAN Fabric High-Level Architecture: CRB | 10
- Campus EVPN-VXLAN Fabric High-Level Architecture: ERB | 11
- Campus IP Clos Fabric High-Level Architecture | 12

Overview of EVPN-VXLAN Fabric With CRB | 14

- Campus EVPN-VXLAN Fabric High Level Architecture: CRB | 14

How to Configure an EVPN-VXLAN Fabric for a Campus Network With CRB | 16

- Requirements | 16
- Overview | 17
- Configure the Underlay | 19
  - Underlay Topology | 19
  - Underlay Configuration | 20
- Configure the Overlay | 22
  - Overlay Topology | 23
  - Overlay and Virtual Network Configuration | 23
- Configure the SRX Series Device | 28
  - SRX Topology | 28
  - SRX Configuration | 29
- Configure Internet Access | 32
  - Internet Access Topology | 33
  - Internet Access Configuration | 33

Configure the Access Layer | 36

Access Layer Topology | 36

Access Layer Configuration | 37

Verification | 40

Configuring Optional Add-Ins to an EVPN-VXLAN Fabric With CRB | 51

How to Configure DHCP | 52

Requirements | 52

Overview | 52

Configuration | 54

Verify DHCP | 55

How to Configure Loop Protection | 60

Requirements | 60

Overview | 60

Configuration | 61

Verify Loop Protection | 62

EVPN-VXLAN Campus Network Scaling Data for CRB | 63

Appendix: Full Device Configurations | 65

Core Device Configurations | 65

Distribution Device Configurations | 72

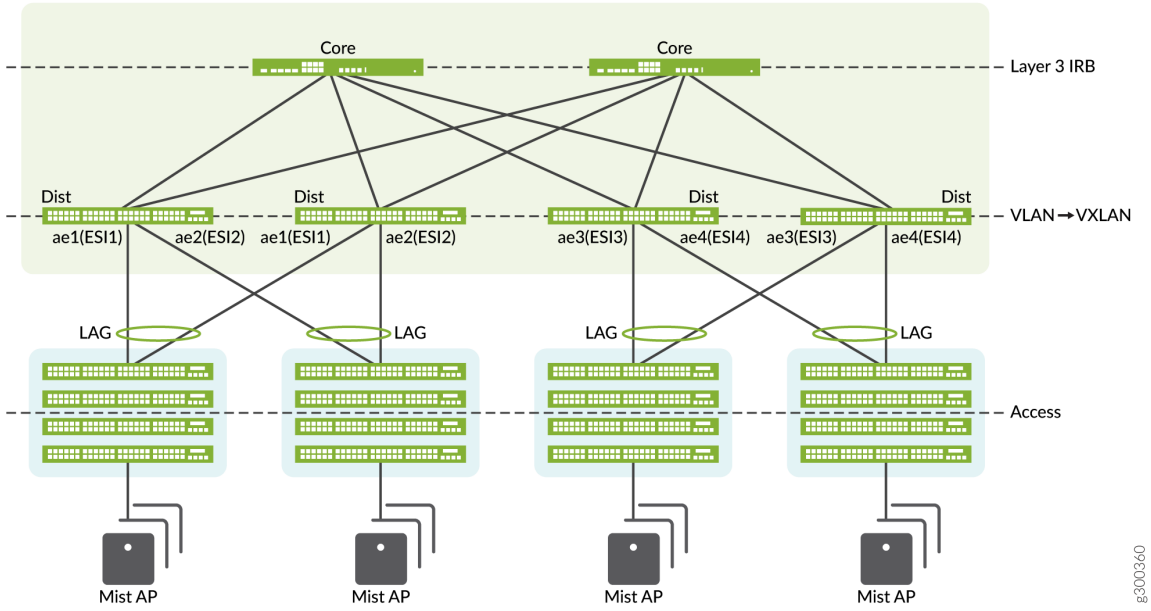
Access Device Configurations | 75

MX Series Router Configuration | 76

SRX Series Device Configuration | 76

About This Network Configuration Example

Use this network configuration example (NCE) to deploy a campus fabric with a Layer 3 IP-based underlay with EVPN as the control plane protocol and VXLAN for data plane encapsulation in the overlay network. In this example, you deploy a centrally-routed bridging (CRB) architecture.



See EVPN-VXLAN Campus Architectures for details on supported EVPN-VXLAN campus architectures. Refer to "Technology Primer: EVPN-VXLAN Fabrics for the Campus" on page 2 for background information on EVPN-VXLAN technology benefits for a campus network.

CHAPTER 1

Example: How to Configure an EVPN-VXLAN Fabric for a Campus Network using CRB

Technology Primer: EVPN-VXLAN Fabrics for the Campus | 2

Overview of EVPN-VXLAN Fabric With CRB | 14

How to Configure an EVPN-VXLAN Fabric for a Campus Network With CRB | 16

Configuring Optional Add-Ins to an EVPN-VXLAN Fabric With CRB | 51

EVPN-VXLAN Campus Network Scaling Data for CRB | 63

Appendix: Full Device Configurations | 65

Technology Primer: EVPN-VXLAN Fabrics for the Campus

IN THIS SECTION

- Overview of EVPN-VXLAN-Based Campus Networks | 2
- EVPN-VXLAN Technical Overview | 4
- Campus EVPN-VXLAN Fabric High-Level Architecture: CRB | 10
- Campus EVPN-VXLAN Fabric High-Level Architecture: ERB | 11
- Campus IP Clos Fabric High-Level Architecture | 12

Overview of EVPN-VXLAN-Based Campus Networks

IN THIS SECTION

- Need for an Overlay-Based Campus Fabric | 2
- EVPN-VXLAN Benefits | 3

Need for an Overlay-Based Campus Fabric

Enterprise networks are adopting cloud-based applications to improve their competitiveness, lower IT costs, and provide users with anytime, anywhere access to resources and data. Mobile devices, social media, and collaboration tools place new demands on networks. Modern enterprise networks must scale rapidly and provide immediate access to devices with limited networking capabilities as the use of Internet of Things (IoT) devices increases.

Most traditional campus architectures use single-vendor, chassis-based technologies that work well in small, static campuses with few endpoints. However, they are too rigid to support the scalability and changing needs of modern large enterprises.

The Juniper Networks EVPN-VXLAN fabric is a highly scalable architecture that is simple, programmable, and built on a standards-based architecture that is common across campuses and data centers.

The EVPN-VXLAN campus architecture uses a Layer 3 IP-based underlay network and an EVPN-VXLAN overlay network. The simple IP-based Layer 3 network underlay limits the Layer 2 broadcast domain. A flexible overlay network based on a VXLAN overlay with an EVPN control plane efficiently provides Layer 3 or Layer 2 connectivity.

This architecture decouples the virtual topology from the physical topology, which improves network flexibility and simplifies network management. Endpoints that require Layer 2 adjacency, such as IoT devices, can be placed anywhere in the network and remain connected to the same logical Layer 2 network.

With an EVPN-VXLAN campus architecture, you can easily add core, distribution, and access layer devices as your business grows without having to redesign the network. EVPN-VXLAN is vendor-agnostic, so you can use the existing access layer infrastructure and gradually migrate to access layer switches that support EVPN-VXLAN capabilities.

EVPN-VXLAN Benefits

An EVPN-VXLAN fabric is an efficient and scalable way to build and connect campus, data center, and public cloud networks. With a robust BGP/EVPN implementation on all platforms, this architecture provides optimized, seamless, and standards-compliant Layer 2 or Layer 3 connectivity.

Juniper Networks EVPN-VXLAN campus networks provide the following benefits:

- *Consistent, scalable architecture*—Enterprises typically have multiple sites with different size requirements. A common EVPN-VXLAN-based campus architecture is consistent across all sites, irrespective of the size. EVPN-VXLAN scales out or scales in as a site evolves.
- *Multi-vendor deployment*—The EVPN-VXLAN architecture uses standards-based protocols so enterprises can deploy campus networks using multi-vendor network equipment. There is no single vendor lock-in requirement.
- *Reduced flooding and learning*—Control plane-based Layer 2/Layer 3 learning reduces the flood and learn issues associated with data plane learning. Learning MAC addresses in the forwarding plane has an adverse impact on network performance as the number of endpoints grows. The EVPN control plane handles the exchange and learning of routes, so newly learned MAC addresses are not exchanged in the forwarding plane.
- *Location-agnostic connectivity*—The EVPN-VXLAN campus architecture provides a consistent endpoint experience no matter where the endpoint is located. Some endpoints require Layer 2 reachability, such as legacy building security systems or IoT devices. The Layer 2 VXLAN overlay provides Layer 2 reachability across campuses without any changes to the underlay network. With

our standards-based network access control integration, an endpoint can be connected anywhere in the network.

- *Underlay agnostic*—VXLAN as an overlay is underlay agnostic. With a VXLAN overlay, you can connect multiple campuses with a Layer 2 VPN or Layer 3 VPN service from a WAN provider or by using IPsec over Internet.
- *Consistent network segmentation*—A universal EVPN-VXLAN-based architecture across campuses and data centers means consistent end-to-end network segmentation for endpoints and applications.
- *Simplified management*—Campuses and data centers based on a common EVPN-VXLAN design can use common tools and network teams to deploy and manage campus and data center networks.

EVPN-VXLAN Technical Overview

IN THIS SECTION

- Understanding VXLAN | 4
- VXLAN Control Plane Limitations | 5
- Understanding EVPN | 5
- The Underlay Network | 6
- The Overlay Network Control Plane | 7
- The Overlay Data Plane | 8
- Access Layer | 8
- Mist Access points | 9
- VRF Segmentation | 9

Understanding VXLAN

Network overlays are created by encapsulating traffic and tunneling it over a physical network. The Virtual Extensible LAN (VXLAN) tunneling protocol encapsulates Layer 2 Ethernet frames in Layer 3 UDP packets. VXLAN enables virtual Layer 2 subnets or segments that can span the underlying physical Layer 3 network.

In a VXLAN overlay network, each Layer 2 subnet or segment is uniquely identified by a virtual network identifier (VNI). A VNI segments traffic the same way that a VLAN ID segments traffic. As is the case

with VLANs, endpoints within the same virtual network can communicate directly with each other. Endpoints in different virtual networks require a device that supports inter-VXLAN routing, which is typically a router or a high-end switch.

The entity that performs VXLAN encapsulation and decapsulation is called a VXLAN tunnel endpoint. Each VXLAN tunnel endpoint is typically assigned a unique IP address.

VXLAN Control Plane Limitations

VXLAN can be deployed as a tunneling protocol across a Layer 3 IP fabric data center without a control plane protocol. The VXLAN abstraction does not change the flood and learn behavior of the Ethernet protocol, which has inherent limitations in terms of scalability and efficiency.

The two primary methods for using VXLAN without a control plane protocol—static unicast VXLAN tunnels and VXLAN with a multicast underlay—do not solve the inherent flood and learn problem and are difficult to scale in large multitenant environments. EVPN-VXLAN is a scalable solution for the flood and learn problems with Ethernet.

Understanding EVPN

Ethernet VPN (EVPN) is a standards-based protocol that provides virtual multipoint bridged connectivity between different domains over an IP or IP/MPLS backbone network. EVPN enables seamless multitenant, flexible services that can be extended on demand.

EVPN is an extension to BGP that allows the network to carry both Layer 2 MAC and Layer 3 IP information simultaneously to optimize routing and switching decisions. This control plane technology uses Multiprotocol BGP (MP-BGP) for MAC and IP address endpoint distribution, where MAC addresses are treated as routes. EVPN enables devices acting as virtual tunnel endpoints (VTEPs) to exchange reachability information with each other about their endpoints.

EVPN provides multipath forwarding and redundancy through an all-active model. The access layer can connect to two or more distribution devices and forward traffic using all of the links. If an access link or distribution device fails, traffic flows from the access layer toward the distribution layer using the remaining active links. For traffic in the other direction, remote distribution devices update their forwarding tables to send traffic to the remaining active distribution devices connected to the multihomed Ethernet segment.

The benefits of using EVPNs include:

- MAC address mobility
- Multitenancy
- Load balancing across multiple links
- Fast convergence

The technical capabilities of EVPN include:

- Minimal flooding—EVPN creates a control plane that shares end host MAC addresses between VTEPs in the same EVPN segment, which minimizes flooding and facilitates MAC address learning.
- Multihoming—EVPN supports multihoming for client devices. A control protocol like EVPN that enables synchronization of endpoint addresses between the distribution switches is needed to support multihoming, because traffic traveling across the topology needs to be intelligently moved across multiple paths.
- Aliasing—EVPN leverages all-active multihoming to allow a remote distribution device to load-balance traffic across the network toward the access layer.
- Split horizon—Split horizon prevents the looping of broadcast, unknown unicast, and multicast (BUM) traffic in a network. With split horizon, a packet is never sent back in the direction it came from.

The Underlay Network

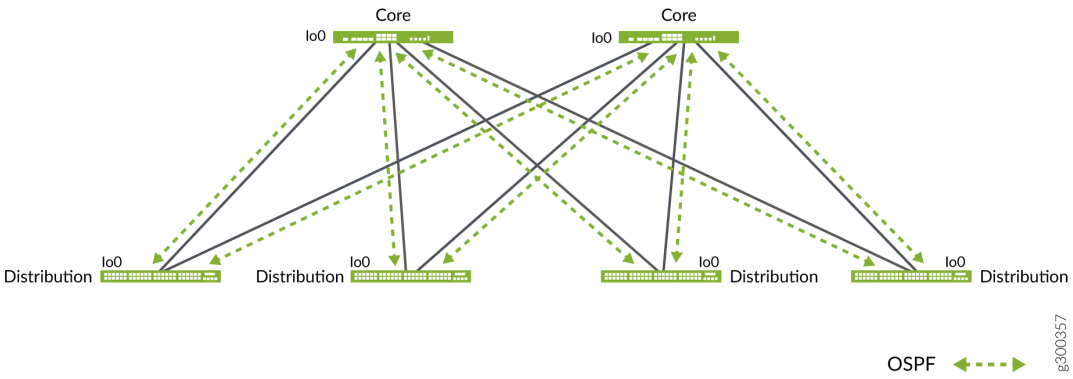
An EVPN-VXLAN fabric architecture makes the network infrastructure simple and consistent across campuses and data centers. All the core and distribution devices must be connected to each other using a Layer 3 infrastructure. We recommend deploying a Clos-based IP fabric with a spine-leaf-based topology to ensure predictable performance and to enable a consistent, scalable architecture.

The primary requirement in the underlay network is that all core and distribution devices have loopback reachability to one another. The loopback addresses are used to establish IBGP peering relationships for the overlay network.

You can use any Layer 3 routing protocol to exchange loopback addresses between the core and distribution devices. BGP provides benefits like better prefix filtering, traffic engineering, and traffic tagging, while OSPF is relatively simple to configure and troubleshoot.

We are using eBGP as the underlay routing protocol in this example because of its ease of use. You can also use OSPF as the underlaying routing protocol if you choose to. Figure 1 on page 7 shows the topology of the underlay network.

Figure 1: Underlay Network Topology



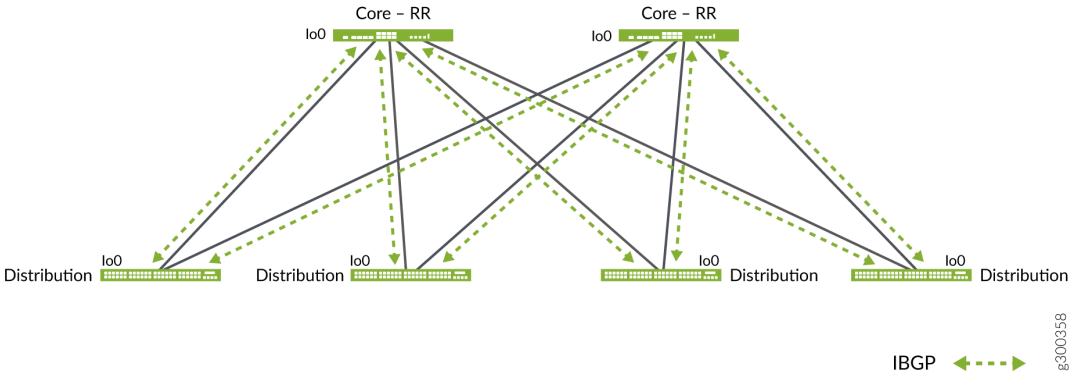
The Overlay Network Control Plane

MP-BGP with EVPN signaling acts as the overlay control plane protocol. The core and distribution devices establish IBGP sessions between each other.

To eliminate the need for full mesh IBGP sessions between all devices, the core switches act as route reflectors and the distribution devices act as route reflector clients. Route reflectors enable simple and consistent IBGP configuration on all distribution switches.

Figure 2 on page 7 shows the topology of the overlay network.

Figure 2: Overlay Network Topology



The Overlay Data Plane

This architecture uses VXLAN as the overlay data plane encapsulation protocol. A Juniper switch that functions as a Layer 2 or Layer 3 VXLAN gateway acts as the VXLAN tunnel endpoint and can encapsulate and decapsulate data packets.

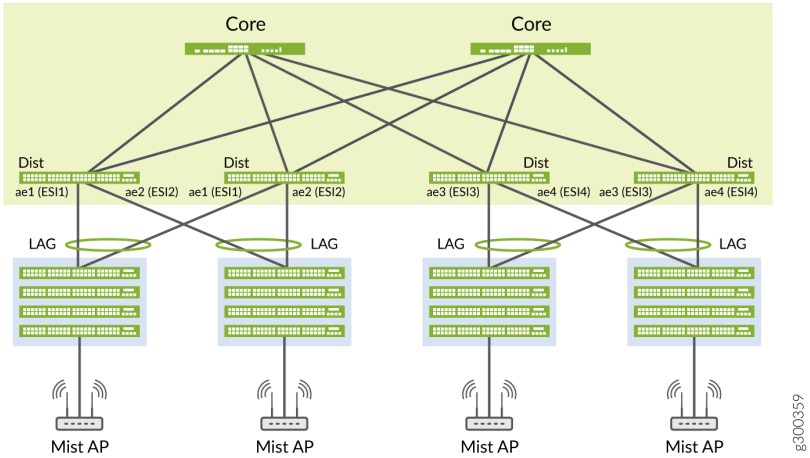
Access Layer

The access layer provides network connectivity to end-user devices, such as personal computers, VoIP phones, printers, IoT devices, as well as connectivity to wireless access point devices. The access layer does not participate in the EVPN-VXLAN fabric and operates at Layer 2 only. The uplinks from the access layer to the distribution layer are Layer 2 trunk link aggregation group (LAG) ports with VLANs relevant to the access switch or Virtual Chassis.

In this example, each access switch or Virtual Chassis is multihomed to two distribution switches. With EVPN running as the control plane protocol, any access switch or Virtual Chassis device can enable active-active multihoming on its interfaces. EVPN provides a standards-based multihoming solution that scales horizontally across any number of distribution layer switches. The access layer switches can use LAG with Link Aggregation Control Protocol (LACP) for multihoming to two more distribution layer switches.

Figure 3 on page 8 shows the topology of the access layer devices after multihoming.

Figure 3: Access Layer Topology



Mist Access points

In our network, we choose Mist Access points as our preferred access point devices. They are designed from the ground up to meet the stringent networking needs of the modern cloud and smart-device era. Mist delivers unique capabilities for both wired and wireless LAN.

- Wired and wireless assurance—Mist is enabled with wired and wireless assurance. Once configured, Service Level Expectations (SLE) for key wired and wireless performance metrics such as throughput, capacity, roaming, and uptime are addressed in the Mist platform. This NCE uses Mist wired assurance services.

- Marvis—An integrated AI engine that provides rapid wired and wireless troubleshooting, trending analysis, anomaly detection, and proactive problem remediation.

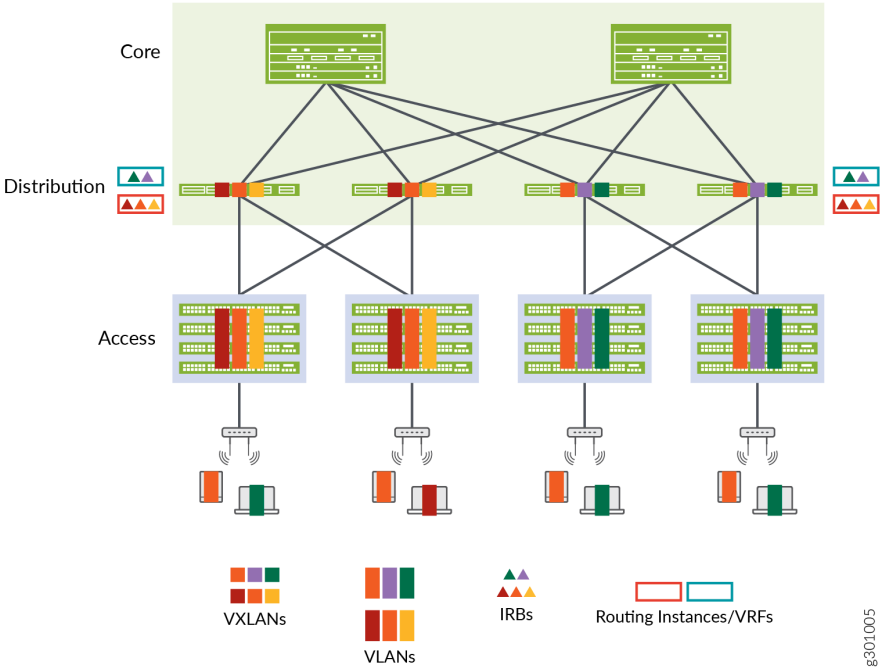
Evolving IT departments look for a cohesive approach for managing wired and wireless networks. Juniper Networks has a solution that simplifies and automate operations, provides end-to-end troubleshooting, and ultimately evolves into the Self-Driving Network™. The Integration of the Mist platform in this NCE addresses both of these challenges. For more details on Mist integration and EX switches, see How to Connect Mist Access Points and Juniper EX Series Switches.

VRF Segmentation

Virtual routing and forwarding (VRF) segmentation is used to organize users and devices in groups on a shared network while separating and isolating the different groups. The routing devices on the network create and maintain separate VRF table for each group. The users and devices in a group are placed in one VRF segment and can communicate with each other, but they cannot communicate with users in another VRF segment. If you want to send and receive traffic from one VRF segment to another VRF segment, then you must configure the routing path.

Figure 4 on page 10 shows an ERB-based EVPN-VXLAN campus network with three VRF segments (employees, guests, and IoT devices).

Figure 4: VRF Segmentation in an ERB Architecture



Campus EVPN-VXLAN Fabric High-Level Architecture: CRB

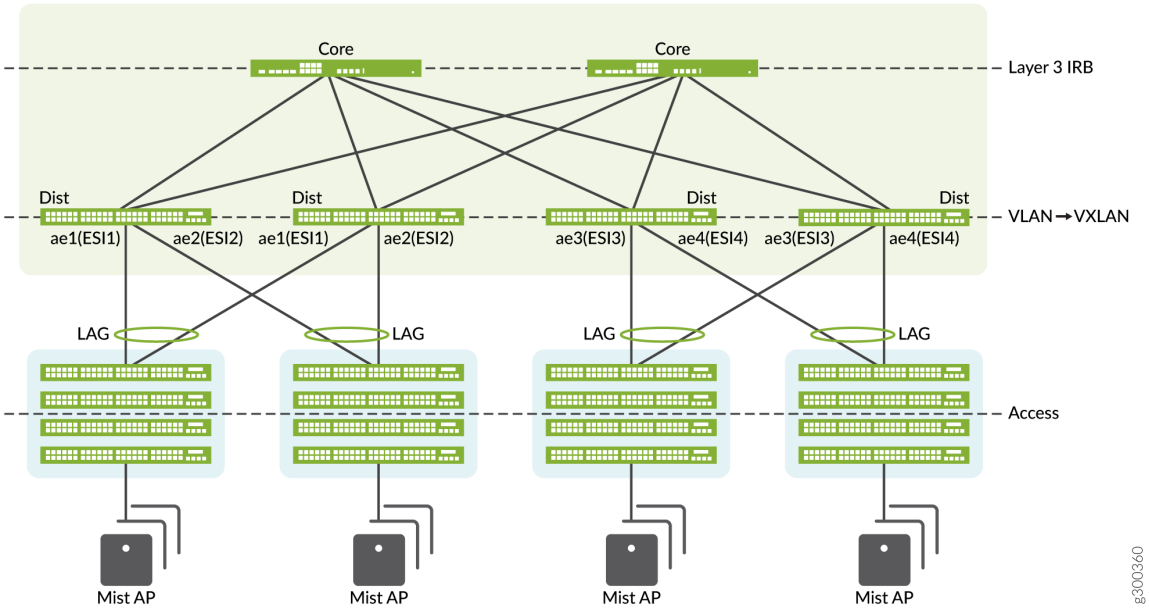
At a high level, a centrally-routed bridging (CRB)-based EVPN-VXLAN fabric architecture for campus network deployments consists of the following:

- Core switches that can be configured as Layer 2/Layer 3 VXLAN gateways.
- A CRB overlay where the integrated routing and bridging (IRB) interfaces for the virtual networks are located on the core switches.
- Distribution switches that can be configured as Layer 2 VXLAN gateways.
- Access layer switches that are either standalone switches or a Virtual Chassis. These switches can be Juniper or third-party devices.

- VLANs that carry endpoint traffic from the wired and wireless devices that connect to the access layer switches and the distribution layer switches.

Figure 5 on page 11 provides a high-level overview of the EVPN-VXLAN fabric architecture for wired and wireless integration.

Figure 5: The CRB EVPN-VXLAN Campus Network Architecture



Campus EVPN-VXLAN Fabric High-Level Architecture: ERB

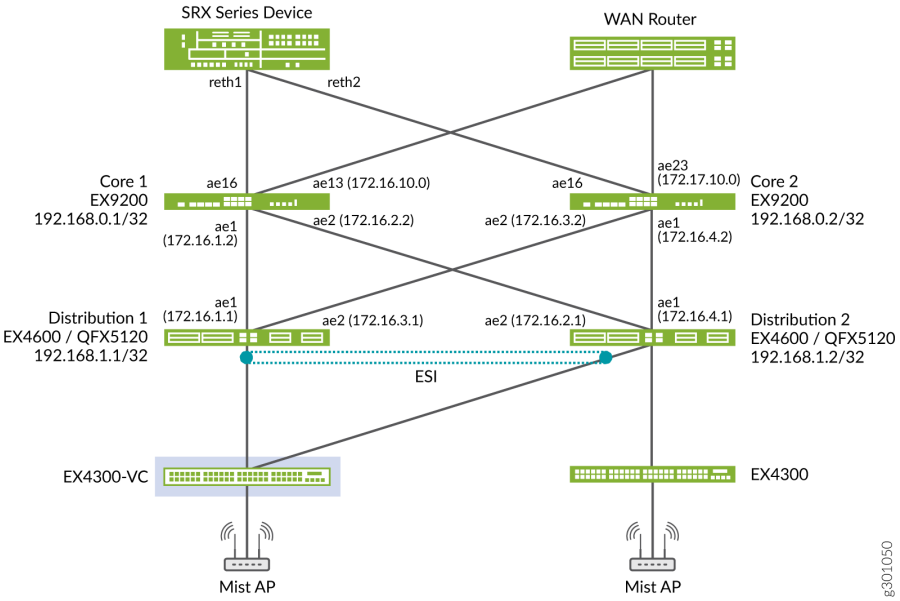
At a high level, an edge-routed bridging (ERB)-based EVPN-VXLAN fabric architecture for campus network deployments consists of the following:

- Core switches provides transport of EVPN Type 2 and Type 5 routes
- Distribution switches can be configured as Layer 2/Layer 3 VXLAN gateways.
- An ERB overlay where the integrated routing and bridging (IRB) interfaces for the virtual networks are located on the distribution switches.
- Access layer switches that are either standalone switches or a Virtual Chassis. These switches can be Juniper or third-party devices.

- VLANs that carry endpoint traffic from the wired and wireless devices that connect to the access layer switches and the distribution layer switches.
- ERB design also enables faster server-to-server, intra-campus traffic (also known as east-west traffic). As a result, routing happens much closer to the end systems than with centrally-routed bridging (CRB) overlays.

  Figure 6 on page 12 shows an EVPN-VXLAN campus network based on the ERB architecture.

Figure 6: The ERB EVPN-VXLAN Campus Network Architecture



Campus IP Clos Fabric High-Level Architecture

The campus fabric, with an EVPN-VXLAN architecture, decouples the overlay network from the underlay network. This approach addresses the needs of the modern enterprise network by allowing network administrators to create logical Layer 2 networks across different Layer 3 networks. By configuring different routing instances, you can create separate virtual networks and each routing instance will have its own separate routing and switching table.

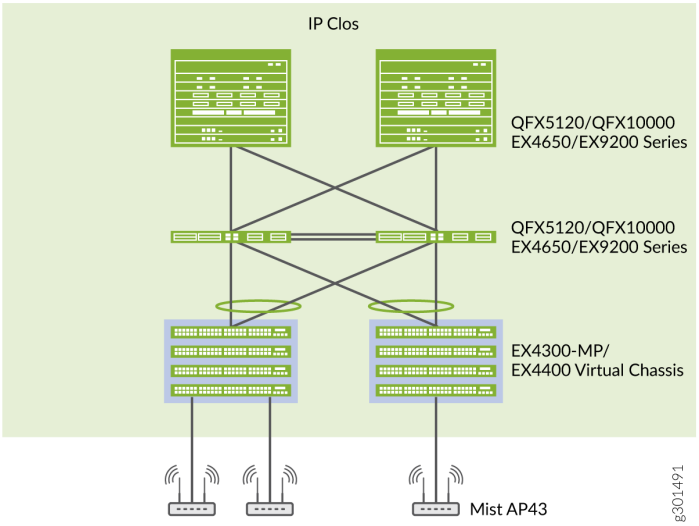
VXLAN is the overlay data plane encapsulation protocol that tunnels Ethernet frames between network endpoints on the Layer 3 IP network. Devices that perform VXLAN encapsulation and decapsulation for the network are referred to as a VXLAN tunnel endpoint (VTEP). Before a VTEP sends a frame into a VXLAN tunnel, it wraps the original frame in a VXLAN header that includes a virtual network identifier

(VNI). The VNI maps the packet to the original VLAN that at the ingress switch. After applying a VXLAN header, the frame is encapsulated into a UDP/IP packet for transmission to another VTEP over an IP network.

A campus fabric with EVPN-VXLAN is a more modern and scalable network that uses a BGP or OSPF underlay from the core to the access layer switches. The access layer switches are VTEPs that encapsulate and decapsulate the VXLAN traffic. In addition, the VTEPs route packets in and out of VXLAN tunnels.

Figure 7 on page 13 shows an EVPN-VXLAN Campus network based on the IP Clos architecture.

Figure 7: An IP Clos Topology



RELATED DOCUMENTATION

- EVPN-VXLAN Campus Architectures
- *Understanding VXLANs*
- EVPN User Guide

Overview of EVPN-VXLAN Fabric With CRB

IN THIS SECTION

- Campus EVPN-VXLAN Fabric High Level Architecture: CRB | 14

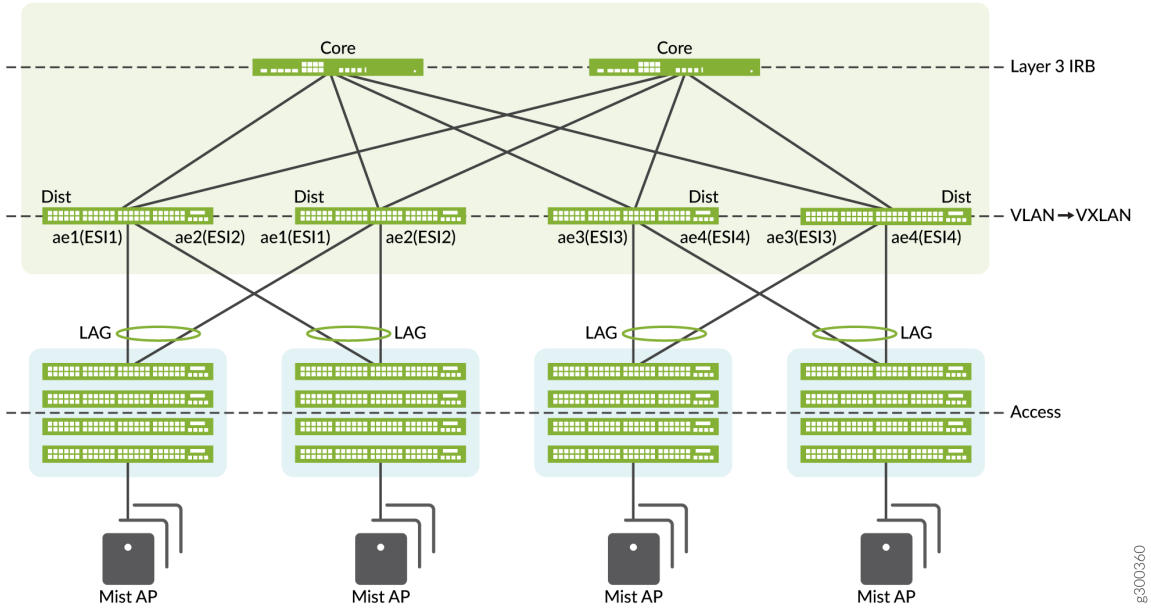
Campus EVPN-VXLAN Fabric High Level Architecture: CRB

At a high level, an EVPN-VXLAN fabric architecture for campus network deployments consists of the following:

- Core switches that can be configured as Layer 2/Layer 3 VXLAN gateways.
- A centrally-routed bridging (CRB) overlay where the integrated routing and bridging (IRB) interfaces for the virtual networks are located on the core switches.
- Distribution switches that can be configured as Layer 2 VXLAN gateways.
- Access layer switches that are either standalone switches or a Virtual Chassis. These switches can be Juniper or third-party devices.
- VLANs that carry endpoint traffic from the wired and wireless devices that connect to the access layer switches and the distribution layer switches.

Figure 8 on page 15 provides a high-level overview of the CRB-based EVPN-VXLAN fabric architecture for wired and wireless integration.

Figure 8: Campus EVPN-VXLAN Architecture



For background information and a technology primer on EVPN-VXLAN for campus networks, see "Overview of EVPN-VXLAN-Based Campus Networks" on page 2.

RELATED DOCUMENTATION

- How to Configure an EVPN-VXLAN Fabric for a Campus Network With ERB

How to Configure an EVPN-VXLAN Fabric for a Campus Network With CRB

IN THIS SECTION

- Requirements | 16
- Overview | 17
- Configure the Underlay | 19
- Configure the Overlay | 22
- Configure the SRX Series Device | 28
- Configure Internet Access | 32
- Configure the Access Layer | 36
- Verification | 40

Requirements

This configuration example uses the following devices:

- Two EX9251 switches as core devices. Software version: Junos OS Release 18.4R2-S4.5.
- Two EX4600 switches as distribution devices. Software version: Junos OS Release 18.4R2-S4.5.
- Two EX4300 switches as access layer devices.
- One SRX Series security device for inter-routing instance traffic inspection.
- One MX Series router for Internet access.
- Three hosts to represent servers.
  - Re-validated using Junos OS Release 21.2R.3.
  - See the Feature Explorer for supported platforms.

Overview

IN THIS SECTION

- Topology | 17

Use this NCE to deploy a single campus fabric with a Layer 3 IP-based underlay network that uses EVPN as the control plane protocol and VXLAN as the data plane protocol in the overlay network. In this example, you deploy an centrally-routed bridging (CRB) architecture with a virtual gateway address (VGA). See EVPN-VXLAN Campus Architectures for details on supported EVPN-VXLAN campus architectures.

First, you configure OSPF as the underlay routing protocol to exchange loopback routes. You then configure IBGP between the core and distribution devices in the overlay to share reachability information about endpoints in the fabric.

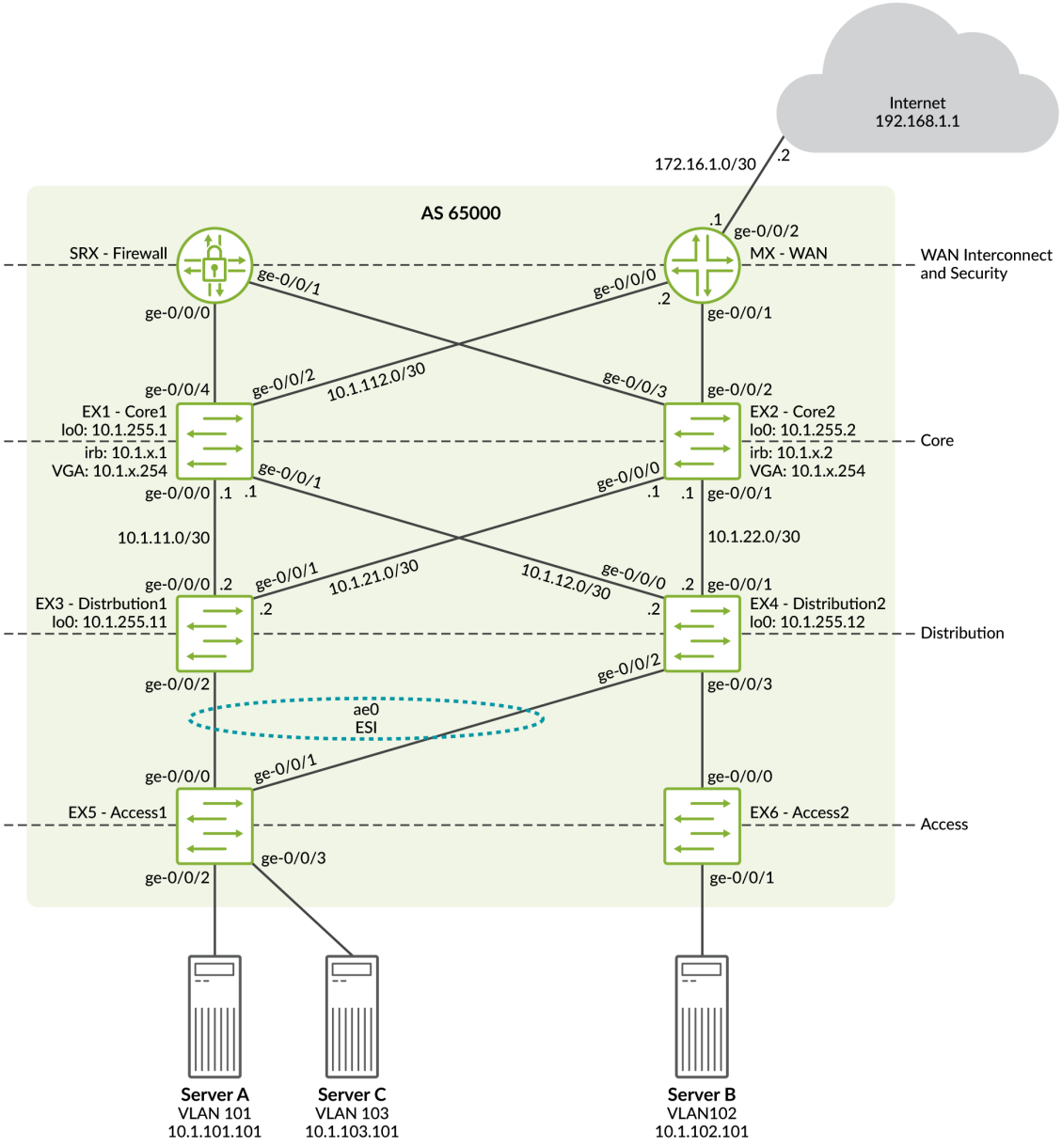
Topology

In this example, both core devices are configured with a unique IRB unicast address along with the same VGA. Figure 9 on page 18 shows the physical topology with a SRX Series device, WAN router, and access layer devices. The IP addressing scheme is also shown.

There are three IRB interfaces to coincide with each VLAN (101, 102, and 103). The IRB interfaces are placed in separate routing instances for network segmentation. The SRX Series device enforces policy rules for transit traffic between Servers A and B. Server A can reach the Internet by leaking a default route learned from the WAN router into the routing instance associated with Server A. Servers B and C

can reach each other directly by using the `auto-export` option to copy routes between their respective routing instances.

Figure 9: EVPN-VXLAN Fabric



NOTE: We use /30 address ranges in this example for readability. If you need to conserve IP address space, consider using /31 addresses.

Also, it's best practice to design the network so the servers can send maximum sized frames without requiring fragmentation. Ethernet does not support fragmentation. Exceeding the core maximum transmission unit (MTU) therefore results in silent discards. To ensure MTU-related drops do not occur, the fabric should support the largest frame that the servers can generate with the added VXLAN encapsulation overhead. This example leaves the servers at the default 1500-byte MTU while configuring the fabric to support a 9000-byte MTU.

Refer to "Appendix: Full Device Configurations" on page 65 for the full configuration of all devices used in the example topology.

Configure the Underlay

IN THIS SECTION

- Underlay Topology | 19
- Underlay Configuration | 20

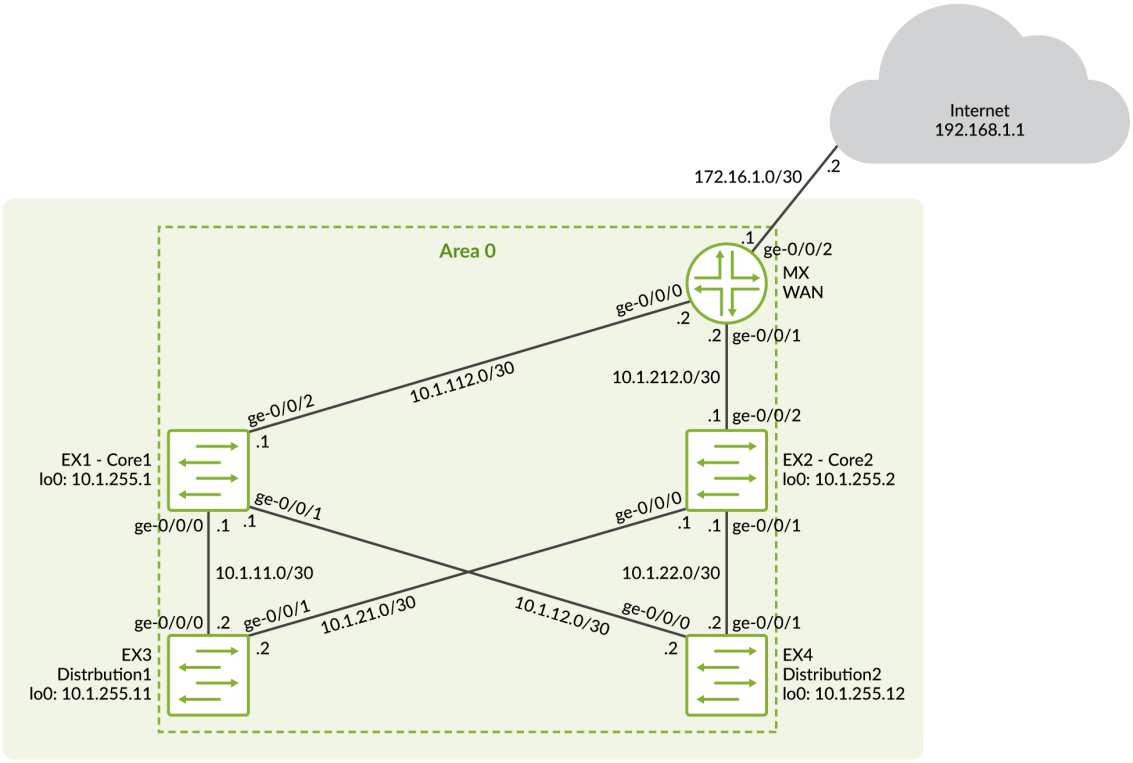
This section shows how to configure the OSPF IP fabric underlay on the core and distribution layer switches.

Underlay Topology

In this example, we use OSPF as the underlay protocol for loopback reachability. The MX Series router is also part of the underlay configuration. Its configuration is detailed in the "Configure Internet Access" on page 32 section of this example.

Figure 10 on page 20 shows the underlay topology.

Figure 10: Underlay Topology



Underlay Configuration

IN THIS SECTION

- Core 1 Configuration | **21**
- Distribution 1 Configuration | **21**

Use this section to configure the underlay on the core and distribution layer switches. We only show the step-by-step configuration for two devices: Core 1 and Distribution 1.

Refer to "Appendix: Full Device Configurations" on page 65 for the full configuration of all devices used in the example topology.

Core 1 Configuration

1. Configure the interfaces connected to the distribution layer switches and the MX Series router.

```
[edit]
user@Core1# set interfaces ge-0/0/0 description to_distribution1
user@Core1# set interfaces ge-0/0/0 mtu 9000
user@Core1# set interfaces ge-0/0/0 unit 0 family inet address 10.1.11.1/30
user@Core1# set interfaces ge-0/0/1 description to_distribution2
user@Core1# set interfaces ge-0/0/1 mtu 9000
user@Core1# set interfaces ge-0/0/1 unit 0 family inet address 10.1.12.1/30
user@Core1# set interfaces ge-0/0/2 description to_mx
user@Core1# set interfaces ge-0/0/2 mtu 9000
user@Core1# set interfaces ge-0/0/2 unit 0 family inet address 10.1.112.1/30
```

2. Configure the loopback interface, the router ID, and per-packet load balancing.

```
[edit]
user@Core1# set interfaces lo0 unit 0 family inet address 10.1.255.1/32
user@Core1# set policy-options policy-statement lb_policy then load-balance per-packet
user@Core1# set policy-options policy-statement lb_policy then accept
user@Core1# set routing-options forwarding-table export lb_policy
user@Core1# set routing-options router-id 10.1.255.1
```

3. Configure the protocol OSPF for the interfaces connected to the distribution layer switches and the MX Series router.

```
[edit]
user@Core1# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
user@Core1# set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
user@Core1# set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 interface-type p2p
user@Core1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Distribution 1 Configuration

1. Configure the interfaces connected to the core devices.

```
[edit]
user@Distribution1# set interfaces ge-0/0/0 description to_core1
```

```
user@Distribution1# set interfaces ge-0/0/0 mtu 9000
user@Distribution1# set interfaces ge-0/0/0 unit 0 family inet address 10.1.11.2/30
user@Distribution1# set interfaces ge-0/0/1 description to_core2
user@Distribution1# set interfaces ge-0/0/1 mtu 9000
user@Distribution1# set interfaces ge-0/0/1 unit 0 family inet address 10.1.21.2/30
```

2. Configure the loopback interface, the router ID, and per-packet load balancing.

```
[edit]
user@Distribution1# set interfaces lo0 unit 0 family inet address 10.1.255.11/32
user@Distribution1# set policy-options policy-statement lb_policy then load-balance per-packet
user@Distribution1# set policy-options policy-statement lb_policy then accept
user@Distribution1# set routing-options forwarding-table export lb_policy
user@Distribution1# set routing-options router-id 10.1.255.11
```

3. Configure the protocol OSPF for the interfaces connected to the core switches.

```
[edit]
user@Distribution1# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
user@Distribution1# set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
user@Distribution1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Configure the Overlay

IN THIS SECTION

- Overlay Topology | 23
- Overlay and Virtual Network Configuration | 23

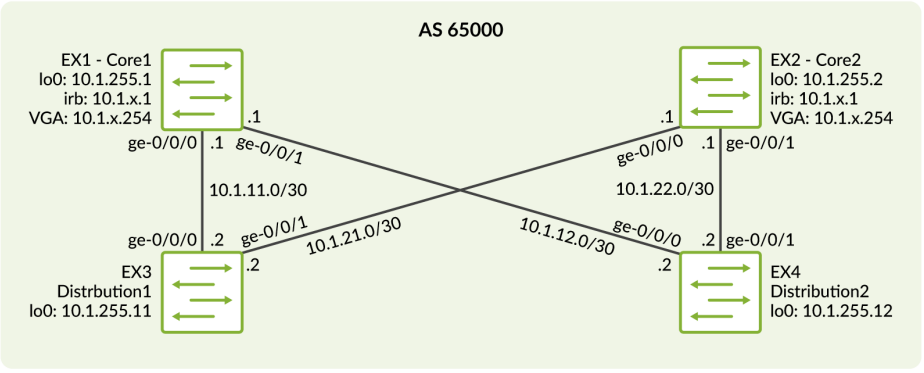
This section shows how to configure the overlay. It includes IBGP peerings, the VLAN to VXLAN mappings, and the IRB interface configurations for the virtual networks (VNs).

Overlay Topology

In this example, there are three VNs: 101, 102 and 103. The IRB interfaces for these VNs are defined on both of the core switches in keeping with a CRB architecture. The IRB interfaces are placed in different routing instance on the core switches for network segmentation. Place the IRB interfaces in the same routing instances if you do not need network segmentation in your deployment.

Figure 11 on page 23 shows the overlay VN topology.

Figure 11: Overlay Topology



x = 101, 102 or 103

Overlay and Virtual Network Configuration

IN THIS SECTION

- Core 1 Configuration | 24
- Distribution 1 Configuration | 27

Use this section to configure the overlay and virtual networks (VNs) on the core and distribution layer switches. We only show the step-by-step configuration for two devices: Core 1 and Distribution 1.

Refer to "Appendix: Full Device Configurations" on page 65 for the full configuration of all devices used in the example topology.

Core 1 Configuration

1. Set the AS number and configure IBGP neighbors between the core and distribution devices.

   You do not need to configure IBGP neighbors between Core 1 and Core 2 because they receive all BGP updates from Distribution 1 and Distribution 2. Configure the core devices as route reflectors to eliminate the need for a full IBGP mesh between all distribution layer switches. Using route reflection makes the configuration on the distribution layer devices simple and consistent.

   NOTE: For full reachability to all IRB interface addresses, add an IBGP peering between Core 1 and Core 2.

```
[edit]
user@Core1# set routing-options autonomous-system 65000
user@Core1# set protocols bgp group OVERLAY type internal
user@Core1# set protocols bgp group OVERLAY local-address 10.1.255.1
user@Core1# set protocols bgp group OVERLAY family evpn signaling
user@Core1# set protocols bgp group OVERLAY cluster 10.1.255.255
user@Core1# set protocols bgp group OVERLAY multipath
user@Core1# set protocols bgp group OVERLAY neighbor 10.1.255.11
user@Core1# set protocols bgp group OVERLAY neighbor 10.1.255.12
```

2. Configure Layer 3 IRB interfaces for the virtual networks and loopback interfaces. The IRB interface units 101, 102 and 103 match the VLAN IDs in our example to represent Servers A, B and C respectively.

```
[edit]
user@Core1# set interfaces irb mtu 9000
user@Core1# set interfaces irb unit 101 proxy-macip-advertisement
user@Core1# set interfaces irb unit 101 virtual-gateway-accept-data
user@Core1# set interfaces irb unit 101 family inet address 10.1.101.1/24 virtual-gateway-
address 10.1.101.254
user@Core1# set interfaces irb unit 101 virtual-gateway-v4-mac 00:12:12:12:12:12
user@Core1# set interfaces irb unit 102 proxy-macip-advertisement
user@Core1# set interfaces irb unit 102 virtual-gateway-accept-data
user@Core1# set interfaces irb unit 102 family inet address 10.1.102.1/24 virtual-gateway-
address 10.1.102.254
user@Core1# set interfaces irb unit 102 virtual-gateway-v4-mac 00:12:12:12:12:12
user@Core1# set interfaces irb unit 103 proxy-macip-advertisement
user@Core1# set interfaces irb unit 103 virtual-gateway-accept-data
```

```
user@Core1# set interfaces irb unit 103 family inet address 10.1.103.1/24 virtual-gateway-
address 10.1.103.254
user@Core1# set interfaces irb unit 103 virtual-gateway-v4-mac 00:12:12:12:12:12
user@Core1# set interfaces lo0 unit 101 family inet address 10.101.255.1/32
user@Core1# set interfaces lo0 unit 102 family inet address 10.102.255.1/32
user@Core1# set interfaces lo0 unit 103 family inet address 10.103.255.1/32
```

3. Configure the overlay virtual networks under a virtual switch routing instance.

```
[edit]
user@Core1# set routing-instances vs_1 instance-type virtual-switch
user@Core1# set routing-instances vs_1 protocols evpn encapsulation vxlan
user@Core1# set routing-instances vs_1 protocols evpn default-gateway no-gateway-community
user@Core1# set routing-instances vs_1 protocols evpn extended-vni-list all
user@Core1# set routing-instances vs_1 vtep-source-interface lo0.0
user@Core1# set routing-instances vs_1 route-distinguisher 10.1.255.1:1
user@Core1# set routing-instances vs_1 vrf-target target:65000:1
user@Core1# set routing-instances vs_1 vrf-target auto
user@Core1# set routing-instances vs_1 vlans v101 vlan-id 101
user@Core1# set routing-instances vs_1 vlans v101 l3-interface irb.101
user@Core1# set routing-instances vs_1 vlans v101 vxlan vni 1101
user@Core1# set routing-instances vs_1 vlans v101 vxlan ingress-node-replication
user@Core1# set routing-instances vs_1 vlans v102 vlan-id 102
user@Core1# set routing-instances vs_1 vlans v102 l3-interface irb.102
user@Core1# set routing-instances vs_1 vlans v102 vxlan vni 1102
user@Core1# set routing-instances vs_1 vlans v102 vxlan ingress-node-replication
user@Core1# set routing-instances vs_1 vlans v103 vlan-id 103
user@Core1# set routing-instances vs_1 vlans v103 l3-interface irb.103
user@Core1# set routing-instances vs_1 vlans v103 vxlan vni 1103
user@Core1# set routing-instances vs_1 vlans v103 vxlan ingress-node-replication
```

4. Configure the VRF routing instance for VLAN 101.

```
[edit]
user@Core1# set routing-instances vrf_101 instance-type vrf
user@Core1# set routing-instances vrf_101 interface irb.101
user@Core1# set routing-instances vrf_101 interface lo0.101
user@Core1# set routing-instances vrf_101 route-distinguisher 10.101.255.1:101
user@Core1# set routing-instances vrf_101 vrf-target target:65000:101
```

5. Configure the VRF routing instance for VLAN 102. We are using a `vrf-import` policy and `routing-options auto-export` to leak routes between the VRFs for VLANs 102 and 103 to allow Servers B and C reachability. The `vrf-import` policy is shown in step 7.

```
[edit]
user@Core1# set routing-instances vrf_102 instance-type vrf
user@Core1# set routing-instances vrf_102 routing-options auto-export
user@Core1# set routing-instances vrf_102 interface irb.102
user@Core1# set routing-instances vrf_102 interface lo0.102
user@Core1# set routing-instances vrf_102 route-distinguisher 10.102.255.1:102
user@Core1# set routing-instances vrf_102 vrf-import serverB_imp
user@Core1# set routing-instances vrf_102 vrf-target target:65000:102
```

6. Configure the VRF routing instance for VLAN 103. We are using a `vrf-import` policy and `routing-options auto-export` to leak routes between the VRFs for VLANs 102 and 103 to allow Servers B and C reachability. The `vrf-import` policy is shown in the next step.

```
[edit]
user@Core1# set routing-instances vrf_103 instance-type vrf
user@Core1# set routing-instances vrf_103 routing-options auto-export
user@Core1# set routing-instances vrf_103 interface irb.103
user@Core1# set routing-instances vrf_103 interface lo0.103
user@Core1# set routing-instances vrf_103 route-distinguisher 10.103.255.1:103
user@Core1# set routing-instances vrf_103 vrf-import serverC_imp
user@Core1# set routing-instances vrf_103 vrf-target target:65000:103
```

7. Configure the `vrf-import` policy to leak routes between the VRFs for VLANs 102 and 103, and apply the policies to the appropriate VRFs as shown in the previous steps. In this example, we are just leaking the IRB interface subnets.

```
[edit]
user@Core1# set policy-options policy-statement serverB_imp term 1 from community serverC
user@Core1# set policy-options policy-statement serverB_imp term 1 from route-filter
10.1.103.0/24 exact
user@Core1# set policy-options policy-statement serverB_imp term 1 then accept
user@Core1# set policy-options policy-statement serverC_imp term 1 from community serverB
user@Core1# set policy-options policy-statement serverC_imp term 1 from route-filter
10.1.102.0/24 exact
user@Core1# set policy-options policy-statement serverC_imp term 1 then accept
```

```
user@Core1# set policy-options community serverB members target:65000:102
user@Core1# set policy-options community serverC members target:65000:103
```

Distribution 1 Configuration

1. Configure IBGP neighbors from the distribution switch to the core switches.

```
[edit]
user@Distribution1# set routing-options autonomous-system 65000
user@Distribution1# set protocols bgp group OVERLAY type internal
user@Distribution1# set protocols bgp group OVERLAY local-address 10.1.255.11
user@Distribution1# set protocols bgp group OVERLAY family evpn signaling
user@Distribution1# set protocols bgp group OVERLAY multipath
user@Distribution1# set protocols bgp group OVERLAY neighbor 10.1.255.1
user@Distribution1# set protocols bgp group OVERLAY neighbor 10.1.255.2
```

2. Configure switch options on the distribution switch.

```
[edit]
user@Distribution1# set switch-options vtep-source-interface lo0.0
user@Distribution1# set switch-options route-distinguisher 10.1.255.11:1
user@Distribution1# set switch-options vrf-target target:65000:1
user@Distribution1# set switch-options vrf-target auto
```

3. Enable VXLAN encapsulation.

```
[edit]
user@Distribution1# set protocols evpn encapsulation vxlan
user@Distribution1# set protocols evpn multicast-mode ingress-replication
user@Distribution1# set protocols evpn extended-vni-list 1101
user@Distribution1# set protocols evpn extended-vni-list 1102
user@Distribution1# set protocols evpn extended-vni-list 1103
```

4. Configure VLANs and VXLAN mappings.

```
[edit]
user@Distribution1# set vlans v101 vlan-id 101
user@Distribution1# set vlans v101 vxlan vni 1101
```

```
user@Distribution1# set vlans v101 vxlan ingress-node-replication
user@Distribution1# set vlans v102 vlan-id 102
user@Distribution1# set vlans v102 vxlan vni 1102
user@Distribution1# set vlans v102 vxlan ingress-node-replication
user@Distribution1# set vlans v103 vlan-id 103
user@Distribution1# set vlans v103 vxlan vni 1103
user@Distribution1# set vlans v103 vxlan ingress-node-replication
```

Configure the SRX Series Device

IN THIS SECTION

- SRX Topology | **28**
- SRX Configuration | **29**

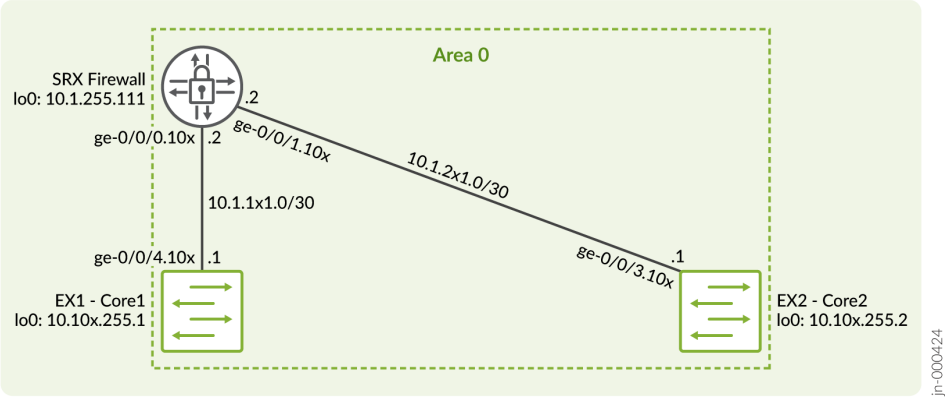
This section shows how to configure the SRX Series device and the core devices to force Servers A and B to communicate through the SRX Series device. In this example, we allow all traffic to pass through the SRX Series device. In a real deployment, you will use the SRX Series device to enforce security policies and possibly other advanced security features like deep packet inspection or unified threat protection (UTP).

SRX Topology

The SRX Series device advertises the routes associated with Servers A and B. The SRX Series device learns these routes from Core 1 and Core 2 using OSPF. This route exchange allows traffic between Servers A and B, but only when the traffic transits the SRX Series device.

Figure 12 on page 29 shows the physical topology.

Figure 12: SRX Series Device Topology



x = 1 or 2

SRX Configuration

IN THIS SECTION

- SRX Series Device Configuration | 29
- Core 1 Configuration | 31

Use this section to configure the SRX Series device, Core 1, and Core 2 to allow Servers A and B to communicate. Only the SRX Series device and Core 1 configurations are shown in the step-by-step procedure. Refer to "Appendix: Full Device Configurations" on page 65 for the full configuration of all devices used in the example topology.

SRX Series Device Configuration

1. Configure the interfaces connected to Core 1 and Core 2. Also configure the loopback interface.

```
[edit]
user@SRX# set interfaces ge-0/0/0 description to_core1
user@SRX# set interfaces ge-0/0/0 vlan-tagging
user@SRX# set interfaces ge-0/0/0 mtu 9000
```

```
user@SRX# set interfaces ge-0/0/0 unit 101 vlan-id 101
user@SRX# set interfaces ge-0/0/0 unit 101 family inet address 10.1.111.2/30
user@SRX# set interfaces ge-0/0/0 unit 102 vlan-id 102
user@SRX# set interfaces ge-0/0/0 unit 102 family inet address 10.1.121.2/30
user@SRX# set interfaces ge-0/0/1 description to_core2
user@SRX# set interfaces ge-0/0/1 vlan-tagging
user@SRX# set interfaces ge-0/0/1 mtu 9000
user@SRX# set interfaces ge-0/0/1 unit 101 vlan-id 101
user@SRX# set interfaces ge-0/0/1 unit 101 family inet address 10.1.211.2/30
user@SRX# set interfaces ge-0/0/1 unit 102 vlan-id 102
user@SRX# set interfaces ge-0/0/1 unit 102 family inet address 10.1.221.2/30
user@SRX# set interfaces lo0 unit 1 family inet address 10.1.255.111/32
```

2. Configure the protocol OSPF to advertise routes between the VRF instances on Core 1 and Core 2. We put the configuration in a routing instance to keep the traffic separate from the main instance.

```
[edit]
user@SRX# set routing-instances v101_v102 instance-type virtual-router
user@SRX# set routing-instances v101_v102 routing-options router-id 10.1.255.111
user@SRX# set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/0.101
interface-type p2p
user@SRX# set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/0.102
interface-type p2p
user@SRX# set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/1.101
interface-type p2p
user@SRX# set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/1.102
interface-type p2p
user@SRX# set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface lo0.1
user@SRX# set routing-instances v101_v102 interface ge-0/0/0.101
user@SRX# set routing-instances v101_v102 interface ge-0/0/0.102
user@SRX# set routing-instances v101_v102 interface ge-0/0/1.101
user@SRX# set routing-instances v101_v102 interface ge-0/0/1.102
user@SRX# set routing-instances v101_v102 interface lo0.1
```

3. Configure the security zones.

```
[edit]
user@SRX# set security zones security-zone v101 host-inbound-traffic system-services ping
user@SRX# set security zones security-zone v101 host-inbound-traffic protocols ospf
user@SRX# set security zones security-zone v101 interfaces ge-0/0/0.101
user@SRX# set security zones security-zone v101 interfaces ge-0/0/1.101
```

```
user@SRX# set security zones security-zone v102 host-inbound-traffic system-services ping
user@SRX# set security zones security-zone v102 host-inbound-traffic protocols ospf
user@SRX# set security zones security-zone v102 interfaces ge-0/0/1.102
user@SRX# set security zones security-zone v102 interfaces ge-0/0/0.102
```

4. Configure the security policies and address books. We configure an "accept all" policy that allows all traffic between the Servers A and B. This is typical for initial testing. After the expected connectivity is verified, you can put detailed security policies into effect.

```
[edit]
user@SRX# set security address-book global address v101 10.1.101.0/24
user@SRX# set security address-book global address v102 10.1.102.0/24
user@SRX# set security policies from-zone v101 to-zone v102 policy v101_to_v102 match source-
address v101
user@SRX# set security policies from-zone v101 to-zone v102 policy v101_to_v102 match
destination-address v102
user@SRX# set security policies from-zone v101 to-zone v102 policy v101_to_v102 match
application any
user@SRX# set security policies from-zone v101 to-zone v102 policy v101_to_v102 then permit
user@SRX# set security policies from-zone v102 to-zone v101 policy v102_to_v101 match source-
address v102
user@SRX# set security policies from-zone v102 to-zone v101 policy v102_to_v101 match
destination-address v101
user@SRX# set security policies from-zone v102 to-zone v101 policy v102_to_v101 match
application any
user@SRX# set security policies from-zone v102 to-zone v101 policy v102_to_v101 then permit
```

Core 1 Configuration

1. Configure the interface connected to the SRX Series device. We configure two units, one for VLAN 101 traffic and one for VLAN 102 traffic.

```
[edit]
user@Core1# set interfaces ge-0/0/4 vlan-tagging
user@Core1# set interfaces ge-0/0/4 mtu 9000
user@Core1# set interfaces ge-0/0/4 unit 101 description v101_to_srx
user@Core1# set interfaces ge-0/0/4 unit 101 vlan-id 101
user@Core1# set interfaces ge-0/0/4 unit 101 family inet address 10.1.111.1/30
user@Core1# set interfaces ge-0/0/4 unit 102 description v102_to_srx
```

```
user@Core1# set interfaces ge-0/0/4 unit 102 vlan-id 102
user@Core1# set interfaces ge-0/0/4 unit 102 family inet address 10.1.121.1/30
```

2. Under the routing instance `vrf_101`, configure the router ID, the OSPF protocol, and the interface associated with VLAN 101.

```
[edit]
user@Core1# set routing-instances vrf_101 routing-options router-id 10.101.255.1
user@Core1# set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface irb.101
passive
user@Core1# set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface lo0.101
user@Core1# set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface ge-0/0/4.101
interface-type p2p
user@Core1# set routing-instances vrf_101 interface ge-0/0/4.101
```

3. Under the routing instance `vrf_102`, configure the router ID, the OSPF protocol, and the interface associated with VLAN 102.

```
[edit]
user@Core1# set routing-instances vrf_102 routing-options router-id 10.102.255.1
user@Core1# set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface irb.102
passive
user@Core1# set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface lo0.102
user@Core1# set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface ge-0/0/4.102
interface-type p2p
user@Core1# set routing-instances vrf_102 interface ge-0/0/4.102
```

Configure Internet Access

IN THIS SECTION

- Internet Access Topology | 33
- Internet Access Configuration | 33

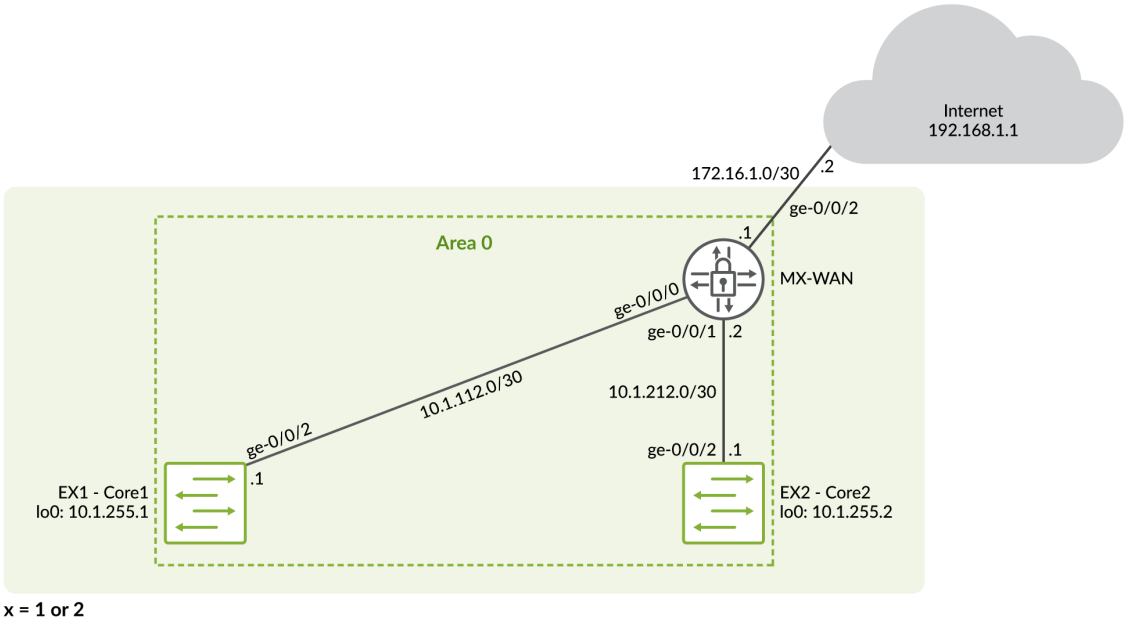
This section shows how to configure the MX Series router and the core devices to allow Server A access to the Internet.

Internet Access Topology

The MX Series router advertises a default route to Core 1 and Core 2 for Internet access. Core 1 and Core 2 copy the default route to the `vrf_101` routing instance associated with Server A. The core devices also advertise the route to reach Server A to the MX Series router.

Figure 13 on page 33 shows the physical topology.

Figure 13: Internet Access Topology



Internet Access Configuration

IN THIS SECTION

- MX Series Router Configuration | 34
- Core 1 Configuration | 35

Use this section to configure Internet access for Server A on Core 1, Core 2, and the MX Series router. Only the MX Series router and Core 1 configurations are shown in the step-by-step procedure. Refer to "Appendix: Full Device Configurations" on page 65 for the full configuration of all devices used in the example topology.

MX Series Router Configuration

1. Configure the interfaces connected to the Core 1 and Core 2 devices. Also configure the loopback interface.

```
[edit]
user@MX# set interfaces ge-0/0/0 description to_core1
user@MX# set interfaces ge-0/0/0 mtu 9000
user@MX# set interfaces ge-0/0/0 unit 0 family inet address 10.1.112.2/30
user@MX# set interfaces ge-0/0/1 description to_core2
user@MX# set interfaces ge-0/0/1 mtu 9000
user@MX# set interfaces ge-0/0/1 unit 0 family inet address 10.1.212.2/30
user@MX# set interfaces ge-0/0/2 unit 0 family inet address 172.16.1.1/30
user@MX# set interfaces lo0 unit 0 family inet address 10.1.255.112/32
```

2. Configure the router ID and OSPF protocol on the core devices to learn and advertise routes.

```
[edit]
user@MX# set routing-options router-id 10.1.255.112
user@MX# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
user@MX# set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
user@MX# set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 passive
user@MX# set protocols ospf area 0.0.0.0 interface lo0.0
user@MX# set protocols ospf export adv-stat
```

3. Configure a static default route and related policy to export the static route into OSPF. We use a static default route in this example, whereas a production network is likely to learn a default route via BGP from the Internet service provider.

```
[edit]
user@MX# set routing-options static route 0.0.0.0/0 next-hop 172.16.1.2
user@MX# set policy-options policy-statement adv-stat term 1 from protocol static
user@MX# set policy-options policy-statement adv-stat term 1 then accept
```

Core 1 Configuration

1. Configure a policy to accept the default route from the MX Series router. This route is then leaked into the routing instance for VLAN 101.

```
[edit]
user@Core1# set policy-options policy-statement default-rt term 1 from route-filter 0.0.0.0/0
exact
user@Core1# set policy-options policy-statement default-rt term 1 then accept
user@Core1# set policy-options policy-statement default-rt term 2 then reject
```

2. Configure a policy to advertise the Server A subnet route to the MX Series router. The route is created in the next step.

```
[edit]
user@Core1# set policy-options policy-statement adv-vrfs term 1 from protocol static
user@Core1# set policy-options policy-statement adv-vrfs term 1 from route-filter
10.1.101.0/24 exact
user@Core1# set policy-options policy-statement adv-vrfs term 1 then accept
```

3. Configure a static route for the Server A subnet with a next routing table of `vrf_101.inet.0`.

   The static route is advertised to the MX Series router. Traffic sent by the MX Series router to Server A uses the `vrf_101` routing instance. We also configure `rib-groups` to import the default route learned from the MX Series router into the `vrf_101` routing instance.

```
[edit]
user@Core1# set routing-options static route 10.1.101.0/24 next-table vrf_101.inet.0
user@Core1# set routing-options rib-groups wan import-rib inet.0
user@Core1# set routing-options rib-groups wan import-rib vrf_101.inet.0
user@Core1# set routing-options rib-groups wan import-policy default-rt
```

4. Apply the `rib-groups` and policy configuration to OSPF.

```
[edit]
user@Core1# set protocols ospf rib-groups inet wan
user@Core1# set protocols ospf export adv-vrfs
```

Configure the Access Layer

IN THIS SECTION

- Access Layer Topology | 36
- Access Layer Configuration | 37

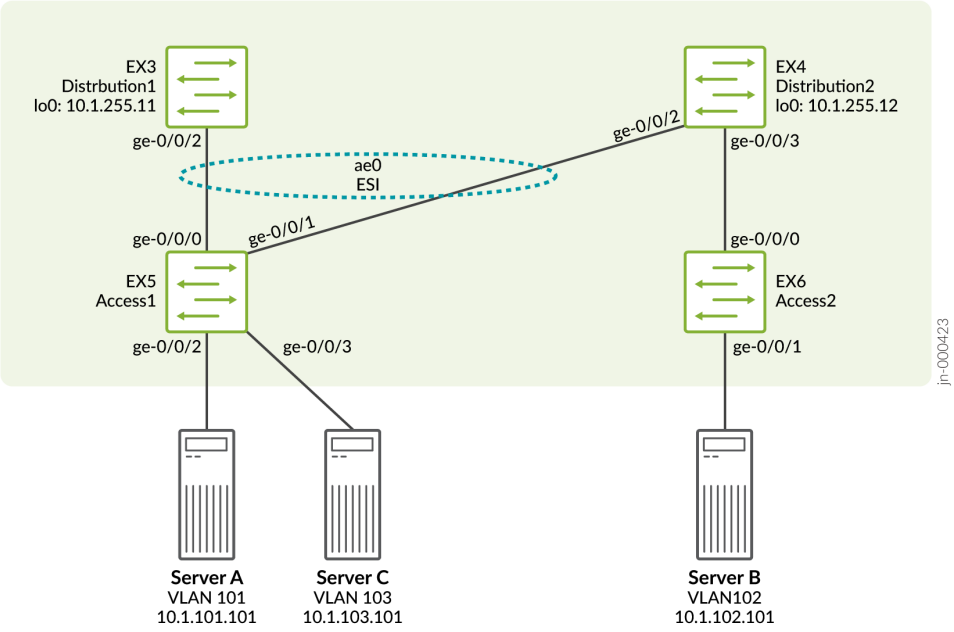
This section shows how to configure multihomed uplink interfaces from an access layer switch to distribution layer devices. The result is an aggregated Ethernet interface that has members connected to multiple distribution layer devices.

Access Layer Topology

The access layer supports Layer 2 for VLANs. The uplink from the access layer is an aggregated Ethernet link bundle, or link aggregation group (LAG). The LAG ae0 is configured as a trunk port to carry traffic from all access layer VLANs to the distribution layer switches.

Figure 14 on page 37 shows the physical topology.

Figure 14: Access Layer Topology



Access Layer Configuration

IN THIS SECTION

- Distribution 1 Configuration | 38
- Access Switch 1 Configuration | 38
- Access Switch 2 Configuration | 39

Use this example to configure the distribution layer for EVPN multihoming. You also configure a conventional LAG interface on the access layer switch.

Only the Distribution 1, Access 1, and Access 2 configurations are shown in the step-by-step procedure. Refer to "Appendix: Full Device Configurations" on page 65 for the Distribution 2 device configuration.

Distribution 1 Configuration

1. Specify the members of the aggregated Ethernet bundle.

```
[edit]
user@Distribution1# set interfaces ge-0/0/2 description to_access1
user@Distribution1# set interfaces ge-0/0/2 ether-options 802.3ad ae0
```

2. Configure the aggregated Ethernet interface. This includes the Ethernet segment identifier (ESI), which assigns multihomed interfaces into an Ethernet segment. The ESI must match on all multihomed interfaces.

```
[edit]
user@Distribution1# set chassis aggregated-devices ethernet device-count 1
user@Distribution1# set interfaces ae0 mtu 9000
user@Distribution1# set interfaces ae0 esi 00:00:01:01:01:01:01:01:01:01
user@Distribution1# set interfaces ae0 esi all-active
user@Distribution1# set interfaces ae0 aggregated-ether-options minimum-links 1
user@Distribution1# set interfaces ae0 aggregated-ether-options lacp active
user@Distribution1# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@Distribution1# set interfaces ae0 aggregated-ether-options lacp system-id
00:00:01:01:01:01
user@Distribution1# set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
user@Distribution1# set interfaces ae0 unit 0 family ethernet-switching vlan members v101
user@Distribution1# set interfaces ae0 unit 0 family ethernet-switching vlan members v102
user@Distribution1# set interfaces ae0 unit 0 family ethernet-switching vlan members v103
```

Access Switch 1 Configuration

1. Specify the link members for the aggregated Ethernet bundle.

```
[edit]
user@Access1# set interfaces ge-0/0/0 description to_dist1
user@Access1# set interfaces ge-0/0/0 ether-options 802.3ad ae0
user@Access1# set interfaces ge-0/0/1 description to_dist2
user@Access1# set interfaces ge-0/0/1 ether-options 802.3ad ae0
```

2. Configure the aggregated Ethernet interface.

```
[edit]
user@Access1# set chassis aggregated-devices ethernet device-count 1
user@Access1# set interfaces ae0 mtu 9000
user@Access1# set interfaces ae0 aggregated-ether-options minimum-links 1
user@Access1# set interfaces ae0 aggregated-ether-options lacp active
user@Access1# set interfaces ae0 aggregated-ether-options lacp periodic fast
user@Access1# set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
user@Access1# set interfaces ae0 unit 0 family ethernet-switching vlan members v101
user@Access1# set interfaces ae0 unit 0 family ethernet-switching vlan members v103
```

3. Configure the VLANs.

```
[edit]
user@Access1# set vlans v101 vlan-id 101
user@Access1# set vlans v103 vlan-id 103
```

4. Configure the interfaces connected to Servers A and C as trunk ports. Servers A and C are tagged in this example, and therefore the `interface-mode` is configured as a trunk.

```
[edit]
user@Access1# set interfaces ge-0/0/2 mtu 9000
user@Access1# set interfaces ge-0/0/2 unit 0 family ethernet-switching interface-mode trunk
user@Access1# set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members v101
user@Access1# set interfaces ge-0/0/3 mtu 9000
user@Access1# set interfaces ge-0/0/3 unit 0 family ethernet-switching interface-mode trunk
user@Access1# set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members v103
```

Access Switch 2 Configuration

1. On Access Switch 2, configure the interface connected to Distribution 2.

```
[edit]
user@Access2# set interfaces ge-0/0/0 mtu 9000
user@Access2# set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk
user@Access2# set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members v102
```

2. Configure the VLANs.

```
[edit]
user@Access2# set vlans v102 vlan-id 102
```

3. Configure the interface connected to Server B as an access port. Server B is untagged in this example, and therefore the `interface-mode` is set to `access`.

```
[edit]
user@Access2# set interfaces ge-0/0/1 mtu 9000
user@Access2# set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access
user@Access2# set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members v102
```

Verification

IN THIS SECTION

- Distribution 1 | 40
- Core 1 | 44
- Reachability | 46

Log in to each device and verify that the EVPN-VXLAN fabric is functional.

Distribution 1

1. On Distribution 1, verify the state of the BGP sessions with the core devices.

   Verify the Distribution 1 device IBGP sessions are established to the loopback addresses of the core devices. Recall the loopbacks are assigned the 10.1.255.1 and 10.1255.2 IP addresses.

```
user@Distribution1> show bgp summary

Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
```

```
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                      52         44          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.1.255.1            65000      35165      31298       0       1 1w2d 19:34:30 Establ
  bgp.evpn.0: 26/26/26/0
  default-switch.evpn.0: 25/25/25/0
  __default_evpn__.evpn.0: 1/1/1/0
10.1.255.2            65000      35808      31814       0       0 1w2d 23:25:30 Establ
  bgp.evpn.0: 18/26/26/0
  default-switch.evpn.0: 18/25/25/0
  __default_evpn__.evpn.0: 0/1/1/0
```

The IBGP sessions are established with the loopback interfaces of the core devices using MP-IBGP. EVPN signaling is correctly enabled to exchange EVPN routes in the overlay.

2. Verify that the EVPN database is correctly populated.

   Verify that the EVPN database is installing MAC address information for locally attached hosts. Confirm the local device is receiving advertisements from the other leaf devices with information about remote hosts.

```
user@Distribution1> show evpn database
Instance: default-switch
VLAN  DomainId  MAC address        Active source                  Timestamp        IP address
     1101       00:12:12:12:12:12  05:00:00:fd:e8:00:00:04:4d:00  Aug 29 15:18:10
10.1.101.254
     1101       2c:6b:f5:2e:2a:f0  10.1.255.1                     Aug 29 15:18:10  10.1.101.1
     1101       2c:6b:f5:ce:a1:f0  10.1.255.2                     Aug 29 11:29:17  10.1.101.2
     1101       56:04:15:00:f9:27  00:00:01:01:01:01:01:01:01:01  Sep 08 10:50:37
10.1.101.101
     1102       00:12:12:12:12:12  05:00:00:fd:e8:00:00:04:4e:00  Aug 29 15:18:10
10.1.102.254
     1102       2c:6b:f5:2e:2a:f0  10.1.255.1                     Aug 29 15:18:10  10.1.102.1
     1102       2c:6b:f5:ce:a1:f0  10.1.255.2                     Aug 29 11:29:17  10.1.102.2
     1102       56:04:15:00:dd:f3  10.1.255.12                    Aug 29 18:10:51
10.1.102.101
     1103       00:12:12:12:12:12  05:00:00:fd:e8:00:00:04:4f:00  Aug 29 15:32:25
10.1.103.254
     1103       2c:6b:f5:2e:2a:f0  10.1.255.1                     Aug 29 17:48:32  10.1.103.1
     1103       2c:6b:f5:ce:a1:f0  10.1.255.2                     Aug 29 17:51:40  10.1.103.2
```

```
     1103       56:04:15:00:b8:6e  00:00:01:01:01:01:01:01:01:01  Sep 08 10:38:55
10.1.103.101
```

The output confirms that the EVPN database is properly learning and installing MAC routes for all endpoints. It also shows the relationship between MAC addresses and their associated VNIs: 1101, 1102 and 1103.

The EVPN database learns MAC addresses with source 00:00:01:01:01:01:01:01:01:01 from the access layer, which is multihomed to the distribution layer. This learning behavior is evidenced by the presence of the ESI—previously configured as 00:00:01:01:01:01:01:01:01:01—as the active source for these entries.

3. Verify that the local switching table is correctly populated.

   Verify that the local switching table is installing MAC address information for locally attached hosts. Also check that it is receiving MAC advertisements from remote leaf devices to learn about remote hosts.

```
user@Distribution1> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
           SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)


Ethernet switching table : 12 entries, 12 learned
Routing instance : default-switch
   Vlan                MAC                 MAC      Logical                 SVLBNH/      Active
   name                address             flags    interface               VENH Index   source
   v101                00:12:12:12:12:12   DRP      esi.596
05:00:00:fd:e8:00:00:04:4d:00
   v101                2c:6b:f5:2e:2a:f0   DRP      vtep.32769
10.1.255.1
   v101                2c:6b:f5:ce:a1:f0   DRP      vtep.32770
10.1.255.2
   v101                56:04:15:00:f9:27   DLR      ae0.0
   v102                00:12:12:12:12:12   DRP      esi.597
05:00:00:fd:e8:00:00:04:4e:00
   v102                2c:6b:f5:2e:2a:f0   DRP      vtep.32769
10.1.255.1
   v102                2c:6b:f5:ce:a1:f0   DRP      vtep.32770
10.1.255.2
   v102                56:04:15:00:dd:f3   DR       vtep.32771
10.1.255.12
```

```
   v103                00:12:12:12:12:12   DRP      esi.611
05:00:00:fd:e8:00:00:04:4f:00
   v103                2c:6b:f5:2e:2a:f0   DRP      vtep.32769
10.1.255.1
   v103                2c:6b:f5:ce:a1:f0   DRP      vtep.32770
10.1.255.2
   v103                56:04:15:00:b8:6e   DL       ae0.0
```

The output confirms the local switching table is correctly learning and installing MAC addresses for all endpoints. It also shows the relationship between MAC addresses and the VLANs they are associated with (in this case, VLANs 101, 102 and 103). A next-hop interface is listed for each MAC. Remote MACs are associated with a VTEP for VXLAN tunneling.

4. Check the multihome connection from Access Switch 1 to the distribution devices. Verify:

   - The local interfaces that are part of the Ethernet segment
   - The remote distribution devices that are part of the same Ethernet segment
   - The bridge domains that are part of the Ethernet segment
   - The designated forwarder for the Ethernet segment

```
user@Distribution1> show evpn instance esi 00:00:01:01:01:01:01:01:01:01 extensive
Instance: default-switch
  Route Distinguisher: 10.1.255.11:1
  Encapsulation type: VXLAN
  Duplicate MAC detection threshold: 5
  Duplicate MAC detection window: 180
  MAC database status                    Local  Remote
    MAC advertisements:                      2      14
    MAC+IP advertisements:                   2      11
    Default gateway MAC advertisements:      0       0
  Number of local interfaces: 2 (2 up)
    Interface name  ESI                            Mode             Status     AC-Role
    .local..5       00:00:00:00:00:00:00:00:00:00  single-homed     Up         Root
    ae0.0           00:00:01:01:01:01:01:01:01:01  all-active       Up         Root
  Number of IRB interfaces: 0 (0 up)
  Number of protect interfaces: 0
  Number of bridge domains: 3
    VLAN  Domain-ID Intfs/up   IRB-intf  Mode            MAC-sync IM-label  MAC-label  v4-SG-
sync IM-core-NH v6-SG-sync IM-core-NH Trans-ID
    101   1101         1  1             Extended        Enabled  1101
Disabled             Disabled             1101
```

```
    102   1102         1  1              Extended        Enabled  1102
Disabled              Disabled              1102
    103   1103         1  1              Extended        Enabled  1103
Disabled              Disabled              1103
  Number of neighbors: 3
    Address               MAC    MAC+IP        AD        IM        ES Leaf-label Remote-DCI-
Peer
    10.1.255.1              6         6         3         3         0
    10.1.255.2              6         6         3         3         0
    10.1.255.12             2         2         2         3         0
  Number of ethernet segments: 4
    ESI: 00:00:01:01:01:01:01:01:01:01
      Status: Resolved by IFL ae0.0
      Local interface: ae0.0, Status: Up/Forwarding
      Number of remote PEs connected: 1
        Remote-PE        MAC-label  Aliasing-label  Mode
        10.1.255.12      1101       0               all-active
      DF Election Algorithm: MOD based
      Designated forwarder: 10.1.255.12
      Backup forwarder: 10.1.255.11
      Last designated forwarder update: Sep 07 02:33:34
  Router-ID: 10.1.255.11
  Source VTEP interface IP: 10.1.255.11
  SMET Forwarding: Disabled
```

Interface ae0.0 is part of this Ethernet segment. The virtual networks 101, 102 and 103 are part of this Ethernet segment. The remote provider edge (PE) or distribution device participating in this Ethernet segment is 10.1.255.12.

In this multihomed Ethernet segment, the local distribution device, Distribution 1, is the designated forwarder for broadcast, unknown unicast, and multicast (BUM) traffic. This means only Distribution 1 forwards BUM traffic into this Ethernet segment.

Core 1

1. On Core 1, verify the BGP sessions with the core and distribution devices.

   Verify that IBGP sessions are established to the loopbacks of the distribution devices.

```
user@Core1> show bgp summary

Threading mode: BGP I/O
Default eBGP mode: advertise - accept, receive - accept
```

```
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.evpn.0
                      20         20          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn State|#Active/
Received/Accepted/Damped...
10.1.255.11           65000      31316      35182       0       0 1w2d 19:41:53 Establ
  bgp.evpn.0: 10/10/10/0
  vs_1.evpn.0: 9/9/9/0
  __default_evpn__.evpn.0: 0/0/0/0
10.1.255.12           65000      35155      31344       0       0 1w2d 19:41:54 Establ
  bgp.evpn.0: 10/10/10/0
  vs_1.evpn.0: 9/9/9/0
  __default_evpn__.evpn.0: 0/0/0/0
```

The IBGP sessions are established to the loopback interfaces of the distribution devices using MP-IBGP. EVPN signaling is configured for EVPN route exchange in the overlay.

2. Verify that the EVPN database is correctly populated.

   Verify that the EVPN database is receiving advertisements from the other distribution devices and is installing MAC address information for devices attached to the access layer. The core devices learn these MAC addresses through EVPN.

```
user@Core1> show evpn database
Instance: vs_1
VLAN  DomainId  MAC address        Active source                  Timestamp        IP address
     1101       00:12:12:12:12:12  05:00:00:fd:e8:00:00:04:4d:00  Aug 30 10:21:07
10.1.101.254
     1101       2c:6b:f5:2e:2a:f0  irb.101                        Aug 29 15:18:10  10.1.101.1
     1101       56:04:15:00:f9:27  00:00:01:01:01:01:01:01:01:01  Sep 08 10:53:24
10.1.101.101
     1102       00:12:12:12:12:12  05:00:00:fd:e8:00:00:04:4e:00  Aug 30 10:21:07
10.1.102.254
     1102       2c:6b:f5:2e:2a:f0  irb.102                        Aug 29 15:18:10  10.1.102.1
     1102       56:04:15:00:dd:f3  10.1.255.12                    Aug 29 18:10:51
10.1.102.101
     1103       00:12:12:12:12:12  05:00:00:fd:e8:00:00:04:4f:00  Aug 30 10:21:07
10.1.103.254
     1103       2c:6b:f5:2e:2a:f0  irb.103                        Aug 29 17:48:32  10.1.103.1
     1103       56:04:15:00:b8:6e  00:00:01:01:01:01:01:01:01:01  Sep 08 08:43:37
10.1.103.101
```

The output confirms the EVPN database is properly learning and installing MAC routes for all endpoints. It also shows the relationship between MAC addresses and the VNIs they are associated with (1101, 1102 and 1103).

3. Verify that the local switching table is correctly populated.

   Verify that the local switching table is receiving advertisements from the other distribution devices and installing MAC address information for devices attached to the access layer.

```
user@Core1> show ethernet-switching table

MAC flags (S - static MAC, D - dynamic MAC, L - locally learned, P - Persistent static
           SE - statistics enabled, NM - non configured MAC, R - remote PE MAC, O - ovsdb MAC)


Ethernet switching table : 3 entries, 3 learned
Routing instance : vs_1
   Vlan                MAC                 MAC      Logical                SVLBNH/      Active
   name                address             flags    interface              VENH Index   source
   v101                56:04:15:00:f9:27   DR       esi.694
00:00:01:01:01:01:01:01:01:01
   v102                56:04:15:00:dd:f3   DR       vtep.32769
10.1.255.12
   v103                56:04:15:00:b8:6e   DR       esi.694
00:00:01:01:01:01:01:01:01:01
```

   The output confirms the local switching table is correctly learning and installing MAC addresses for all endpoints. It also shows the relationship between MAC addresses, the VLANs they are associated with (in this case, VLANs 101, 102 and 103), and their next-hop interface.

Reachability

1. Verify Server A to Server B reachability. Confirm Server A can ping Server B by transiting the SRX Series device.

   Verify the route for Server B is in the `vrf_101` routing instance, and that the route for Server A is in the `vrf_102` routing instance on both core devices.

```
user@Core1> show route table vrf_101 10.1.102.0/24

vrf_101.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
10.1.102.0/24      *[OSPF/10] 2w6d 21:16:10, metric 3
                    >  to 10.1.111.2 via ge-0/0/4.101
```

```
user@Core1> show route table vrf_102 10.1.101.0/24

vrf_102.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[OSPF/10] 2w6d 21:16:36, metric 3
                    >  to 10.1.121.2 via ge-0/0/4.102
```

```
user@Core2> show route table vrf_101 10.1.102.0/24

vrf_101.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.102.0/24      *[OSPF/10] 3w0d 01:35:24, metric 3
                    >  to 10.1.211.2 via ge-0/0/3.101
```

```
user@Core2> show route table vrf_102 10.1.101.0/24

vrf_102.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[OSPF/10] 3w0d 01:35:34, metric 3
                    >  to 10.1.221.2 via ge-0/0/3.102
```

Ping Server B from Server A.

```
user@ServerA> ping 10.1.102.101 count 2
PING 10.1.102.101 (10.1.102.101): 56 data bytes
64 bytes from 10.1.102.101: icmp_seq=0 ttl=61 time=16.002 ms
64 bytes from 10.1.102.101: icmp_seq=1 ttl=61 time=9.812 ms

--- 10.1.102.101 ping statistics ---
```

```
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 9.812/12.907/16.002/3.095 ms
```

Confirm the flow session on the SRX Series device. You might need to leave the pings going to see the flow session on the SRX Series device. By default, flow sessions time out after a few seconds.

```
user@SRX> show security flow session protocol icmp
Session ID: 1207, Policy name: v101_to_v102/6, HA State: Stand-alone, Timeout: 2, Valid
  In: 10.1.101.101/49797 --> 10.1.102.101/1;icmp, Conn Tag: 0x0, If: ge-0/0/1.101, Pkts: 1,
Bytes: 84,
  Out: 10.1.102.101/1 --> 10.1.101.101/49797;icmp, Conn Tag: 0x0, If: ge-0/0/1.102, Pkts: 1,
Bytes: 84,
Total sessions: 1
```

Trace the route from Server A to Server B to confirm the Layer 3 forwarding hops.

```
user@ServerA> traceroute 10.1.102.101
traceroute to 10.1.102.101 (10.1.102.101), 30 hops max, 52 byte packets
 1  10.1.101.2 (10.1.101.2)  6.045 ms 10.1.101.1 (10.1.101.1)  4.849 ms 10.1.101.2
(10.1.101.2)  4.152 ms
 2  10.1.211.2 (10.1.211.2)  5.476 ms 10.1.111.2 (10.1.111.2)  4.562 ms 10.1.211.2
(10.1.211.2)  10.474 ms
 3  10.1.221.1 (10.1.221.1)  6.518 ms  5.770 ms  5.338 ms
 4  10.1.102.101 (10.1.102.101)  9.276 ms  10.033 ms  9.352 ms
```

The outputs above confirms that Server A can ping Server B and that the traffic is transiting the SRX Series device.

NOTE: Server A should not be able to reach Server C with the configuration in this example.

2. Verify Server A can reach the Internet through the MX Series router.

   First, verify the default route is in the `vrf_101` routing instance, and that the route for Server A is in the `inet.0` table with a next hop of the `vrf_101.inet.0` table on both core devices.

```
user@Core1> show route table vrf_101 0/0 exact

vrf_101.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
0.0.0.0/0          *[OSPF/150] 1w5d 00:05:32, metric 0, tag 0
                    >  to 10.1.112.2 via ge-0/0/2.0
```

```
user@Core1> show route protocol static table inet.0 10.1.101.0/24

inet.0: 24 destinations, 25 routes (24 active, 0 holddown, 0 hidden)
Limit/Threshold: 1048576/1048576 destinations
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[Static/5] 2w6d 22:12:21
                        to table vrf_101.inet.0
```

```
user@Core2> show route table vrf_101 0/0 exact

vrf_101.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[OSPF/150] 1w5d 00:10:56, metric 0, tag 0
                    >  to 10.1.212.2 via ge-0/0/2.0
```

```
user@Core2> show route protocol static table inet.0 10.1.101.0/24

inet.0: 24 destinations, 25 routes (24 active, 0 holddown, 0 hidden)
Limit/Threshold: 1048576/1048576 destinations
+ = Active Route, - = Last Active, * = Both

10.1.101.0/24      *[Static/5] 3w0d 02:12:19
                        to table vrf_101.inet.0
```

Then generate pings and traceroutes to the Internet (192.168.1.1 in our example) from Server A.

```
user@ServerA> ping 192.168.1.1 count 2
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: icmp_seq=0 ttl=62 time=5.987 ms
64 bytes from 192.168.1.1: icmp_seq=1 ttl=62 time=5.898 ms

--- 192.168.1.1 ping statistics ---
```

```
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 5.898/5.942/5.987/0.045 ms
```

```
user@ServerA> traceroute 192.168.1.1
traceroute to 192.168.1.1 (192.168.1.1), 30 hops max, 52 byte packets
 1  10.1.101.1 (10.1.101.1)  6.875 ms  4.067 ms  4.004 ms
 2  10.1.212.2 (10.1.212.2)  5.502 ms  5.215 ms  6.766 ms
 3  192.168.1.1 (192.168.1.1)  10.083 ms  9.538 ms  8.178 ms
```

The output confirms that Server A can ping the Internet.

3. Verify Server C to Server B reachability. Confirm Server C can ping Server B.

   First, verify the routes for each server are present in the `vrf_102` and `vrf_103`routing instances.

```
user@Core1> show route table vrf_103 10.1.102.0/24

vrf_103.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.102.0/24      *[Direct/0] 1w5d 03:04:21
                    >  via irb.102
```

```
user@Core1> show route table vrf_102 10.1.103.0/24

vrf_102.inet.0: 16 destinations, 16 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.103.0/24      *[Direct/0] 1w5d 03:04:24
                    >  via irb.103
```

Then generate pings and trace the route to Server B from Server C.

```
user@ServerC> ping 10.1.102.101 count 2
PING 10.1.102.101 (10.1.102.101): 56 data bytes
64 bytes from 10.1.102.101: icmp_seq=0 ttl=63 time=10.477 ms
64 bytes from 10.1.102.101: icmp_seq=1 ttl=63 time=9.947 ms

--- 10.1.102.101 ping statistics ---
```

```
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 9.947/10.212/10.477/0.265 ms
```

```
user@ServerC> traceroute 10.1.102.101
traceroute to 10.1.102.101 (10.1.102.101), 30 hops max, 52 byte packets
 1  10.1.103.1 (10.1.103.1)  4.281 ms  4.418 ms  3.156 ms
 2  10.1.102.101 (10.1.102.101)  7.479 ms  7.402 ms  7.398 ms
```

The output above confirms that Server C can ping Server B.

You have successfully configured your EVPN-VXLAN fabric for a campus network with CRB.

RELATED DOCUMENTATION

- EVPN User Guide
- Juniper Networks EVPN Implementation for Next-Generation Data Center Architectures
- EVPN-VXLAN Campus Architectures
- How to Configure an EVPN-VXLAN Fabric for a Campus Network With ERB

Configuring Optional Add-Ins to an EVPN-VXLAN Fabric With CRB

IN THIS SECTION

- How to Configure DHCP | **52**
- Verify DHCP | **55**
- How to Configure Loop Protection | **60**
- Verify Loop Protection | **62**

This section shows how to configure optional features in an EVPN-VXLAN fabric.

WHAT'S NEXT

Juniper's campus solution, based on a VXLAN overlay with EVPN control plane, is an efficient and scalable way to build and interconnect multiple campuses across a core network. With a robust BGP/EVPN implementation, Juniper is well-positioned to harness the full potential of EVPN technology. For more information on available EVPN features and how to configure them, see EVPN User Guide.

How to Configure DHCP

IN THIS SECTION

- Requirements | 52
- Overview | 52
- Configuration | 54

Requirements

Configure DHCP on the following devices that you configured in the "How to Configure an EVPN-VXLAN Fabric for a Campus Network With CRB" on page 16 configuration example:

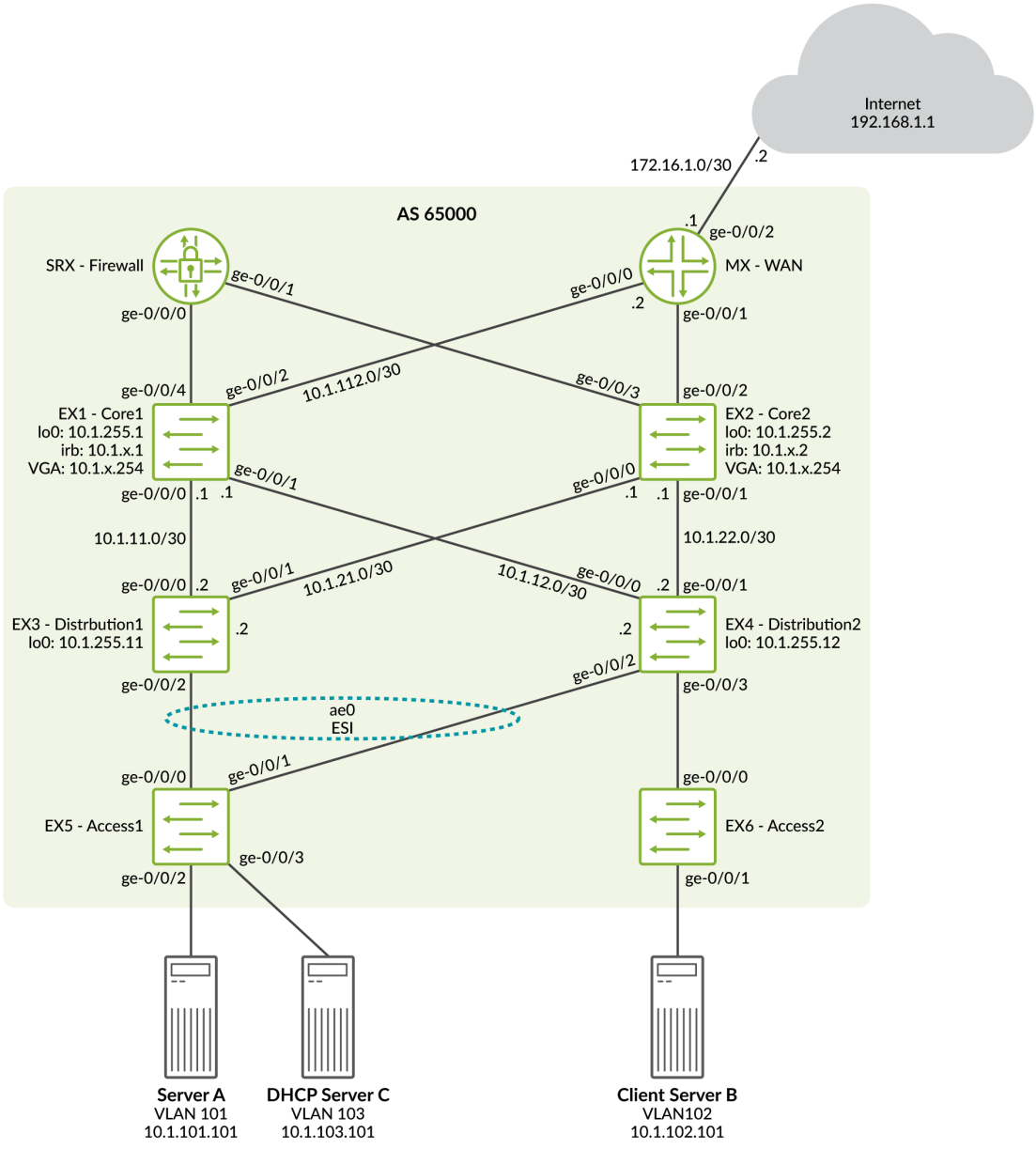
- Two EX9251 switches as core devices running Junos OS Release 18.4R2-S4.5.
- Two EX4600 switches as distribution devices running Junos OS Release 18.4R2-S4.5.
- Two hosts to represent a DHCP server and a client.
  - Re-validated using Junos OS Release 21.2R.3.
  - See the Feature Explorer for supported platforms.

Overview

Use this section to configure DHCP on the network. This example uses segmentation by placing the IRB interfaces in separate routing instances. The DHCP server is connected to one of the IRB interfaces in a service VRF instance. Figure 15 on page 53 shows the virtual network topology with a DHCP server.

To ensure the client gets an IP address from the DHCP server, you can use a device like an SRX Series security device to provide inter-VRF routing, or you can use inter-VRF routing between VRFs locally on the core devices. This example shows how to use inter-VRF routing on the core devices.

Figure 15: Overlay Virtual Network Topology with a DHCP Server



Configuration

We are only showing the DHCP relay configuration steps for the Core 1 device. The steps for the Core 2 device are the same. The inter-VRF routing configuration is shown in steps 5, 6 and 7 of the "Overlay and Virtual Network Configuration" on page 23 example.

1. Configure the DHCP relay option `forward-only-replies` on the core devices under the routing instance `vrf_103`, which is associated with the DHCP server. This DHCP relay option is used when the server and client are in different routing instances.

```
[edit]
user@Core1# set routing-instances vrf_103 forwarding-options dhcp-relay forward-only-replies
```

2. Configure the DHCP relay option `forward-only routing-instance vrf_103` on the core devices under the routing instance `vrf_102`, which is associated with the DHCP client.

```
[edit]
user@Core1# set routing-instances vrf_102 forwarding-options dhcp-relay forward-only routing-
instance vrf_103
```

3. Create a server group under the routing instance `vrf_102` to specify the IP address of the DHCP server.

```
[edit]
user@Core1# set routing-instances vrf_102 forwarding-options dhcp-relay server-group serverC
10.1.103.101
```

4. Configure the server group as the active server group and the interface connected to the client under the routing instance `vrf_102`.

```
[edit]
user@Core1# set routing-instances vrf_102 forwarding-options dhcp-relay group serverC active-
server-group serverC
user@Core1# set routing-instances vrf_102 forwarding-options dhcp-relay group serverC
interface irb.102
```

5. (Optional) Configure DHCP traceoptions.

```
[edit]
user@Core1# set system processes dhcp-service traceoptions file dhcp-test
user@Core1# set system processes dhcp-service traceoptions level all
user@Core1# set system processes dhcp-service traceoptions flag all
```

6. Configure the DHCP server.

> **NOTE**: In this example, we are using an SRX Series device to represent the DHCP server. The host IP address range is between 101 and 200. We provide a default route with the next-hop of the virtual gateway address (VGA) for VLAN 102.

```
[edit]
user@DHCP-ServerC# set system services dhcp-local-server group v101 interface ge-0/0/0.0
user@DHCP-ServerC# set security zones security-zone trust host-inbound-traffic system-
services ping
user@DHCP-ServerC# set security zones security-zone trust host-inbound-traffic system-
services bootp
user@DHCP-ServerC# set security zones security-zone trust interfaces ge-0/0/0.0
user@DHCP-ServerC# set interfaces ge-0/0/0 description to_access1
user@DHCP-ServerC# set interfaces ge-0/0/0 vlan-tagging
user@DHCP-ServerC# set interfaces ge-0/0/0 unit 0 vlan-id 103
user@DHCP-ServerC# set interfaces ge-0/0/0 unit 0 family inet address 10.1.103.101/24
user@DHCP-ServerC# set access address-assignment pool v102 family inet network 10.1.102.0/24
user@DHCP-ServerC# set access address-assignment pool v102 family inet range r1 low
10.1.102.101
user@DHCP-ServerC# set access address-assignment pool v102 family inet range r1 high
10.1.102.200
user@DHCP-ServerC# set access address-assignment pool v102 family inet dhcp-attributes option
3 ip-address 10.1.102.254
user@DHCP-ServerC# set routing-options static route 0.0.0.0/0 next-hop 10.1.103.254
```

Verify DHCP

Log in to the applicable devices and verify DHCP is working.

1. Verify DHCP relay is working on the core devices. Confirm DHCP relay in the traceoptions log.

> **NOTE**: A DHCP relay binding will not be maintained on the core devices. The Core 2 device is the active gateway in this example. You might have to release and renew your IP on the client to see the logs in the traceoptions if you added traceoptions after the client and server configuration. We are only showing a snippet of the traceoptions log.

```
user@Core2> show log dhcp-test
...
Sep 21 01:47:19.438052 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102]
jdhcpd_io_process_ip_packet: SNOOP: recv pkt; sa 10.1.102.101; da 10.1.103.101; src_port 68;
dst_port 67; len 281
Sep 21 01:47:19.438205 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
from == 10.1.102.101, port == 68 ]--
Sep 21 01:47:19.438243 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
size == 281, op == 1 ]--
Sep 21 01:47:19.438258 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
flags == 0 ]--
Sep 21 01:47:19.438274 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
htype == 1, hlen == 6 ]--
Sep 21 01:47:19.438289 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
hops == 0, xid == 19f65ec ]--
Sep 21 01:47:19.438300 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
secs == 0, flags == 0000 ]--
Sep 21 01:47:19.438315 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
ciaddr == 10.1.102.101 ]--
Sep 21 01:47:19.438329 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
yiaddr == 0.0.0.0 ]--
Sep 21 01:47:19.438343 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
siaddr == 0.0.0.0 ]--
Sep 21 01:47:19.438355 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
giaddr == 0.0.0.0 ]--
Sep 21 01:47:19.438387 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
chaddr == 56 04 15 00 dd f3 00 00 00 00 00 00 00 00 00 00 ]--
Sep 21 01:47:19.438399 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
sname ==  ]--
Sep 21 01:47:19.438409 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ DHCP/BOOTP
file ==  ]--
Sep 21 01:47:19.438446 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ OPTION code
55, len  14, data 03 33 01 0f 06 42 43 78 2c 2b 96 0c 07 2a ]--
```

```
Sep 21 01:47:19.438466 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ OPTION code
53, len   1, data DHCP-REQUEST ]--
Sep 21 01:47:19.438490 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ OPTION code
12, len  12, data 56 4d 36 33 30 43 46 42 35 35 41 30 ]--
Sep 21 01:47:19.438502 [MSTR][DEBUG][default:vrf_102][RLY][INET][irb.102] --[ OPTION code
51, len   4, data 00 01 51 80 ]--
Sep 21 01:47:19.438516 [MSTR][INFO] [default:vrf_102][RLY][INET][irb.102] --[ OPTION code
255, len   0 ]--
Sep 21 01:47:19.438640 [MSTR][DEBUG] client_key_compose: Composing key (0x54ac140) for cid_l
0, cid NULL, mac 56 04 15 00 dd f3, htype 1, subnet 10.1.102.2, ifindx 0, opt82_l 0, opt82
NULL
...
Sep 21 01:47:19.464635 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103]
jdhcpd_io_process_ip_packet: SNOOP: recv pkt; sa 10.1.103.101; da 10.1.102.2; src_port 67;
dst_port 67; len 302
Sep 21 01:47:19.464657 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
from == 10.1.103.101, port == 67 ]--
Sep 21 01:47:19.464670 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
size == 302, op == 2 ]--
Sep 21 01:47:19.464694 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
flags == 0 ]--
Sep 21 01:47:19.464706 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
htype == 1, hlen == 6 ]--
Sep 21 01:47:19.464717 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
hops == 0, xid == 19f65ec ]--
Sep 21 01:47:19.464744 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
secs == 0, flags == 0000 ]--
Sep 21 01:47:19.464756 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
ciaddr == 10.1.102.101 ]--
Sep 21 01:47:19.464772 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
yiaddr == 10.1.102.101 ]--
Sep 21 01:47:19.464783 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
siaddr == 0.0.0.0 ]--
Sep 21 01:47:19.464793 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
giaddr == 10.1.102.2 ]--
Sep 21 01:47:19.464808 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
chaddr == 56 04 15 00 dd f3 00 00 00 00 00 00 00 00 00 00 ]--
Sep 21 01:47:19.464817 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
sname ==  ]--
Sep 21 01:47:19.464827 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ DHCP/BOOTP
file ==  ]--
Sep 21 01:47:19.464838 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ OPTION code
53, len   1, data DHCP-ACK ]--
```

```
Sep 21 01:47:19.464851 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ OPTION code
51, len   4, data 00 01 51 80 ]--
Sep 21 01:47:19.464864 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ OPTION code
1, len   4, data ff ff ff 00 ]--
Sep 21 01:47:19.464875 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ OPTION code
54, len   4, data 0a 01 67 65 ]--
Sep 21 01:47:19.464885 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ OPTION code
3, len   4, data 0a 01 66 fe ]--
Sep 21 01:47:19.464905 [MSTR][DEBUG][default:vrf_103][RLY][INET][irb.103] --[ OPTION code
82, len  31, data 09 1d 00 00 0a 4c 18 04 16 49 52 42 2d 69 72 62 2e 31 30 32 3a 76 74 65 70
2e 33 32 37 36 38 ]--
```

2. Verify DHCP relay in the statistics output.

```
user@Core2> show dhcp relay statistics routing-instance vrf_102
Packets dropped:
    Total                     0

Messages received:
    BOOTREQUEST               2
    DHCPDECLINE               0
    DHCPDISCOVER              1
    DHCPINFORM                0
    DHCPRELEASE               0
    DHCPREQUEST               1
    DHCPLEASEACTIVE           0
    DHCPLEASEUNASSIGNED       0
    DHCPLEASEUNKNOWN          0
    DHCPLEASEQUERYDONE        0
    DHCPACTIVELEASEQUERY      0

Messages sent:
    BOOTREPLY                 3
    DHCPOFFER                 2
    DHCPACK                   1
    DHCPNAK                   0
    DHCPFORCERENEW            0
    DHCPLEASEQUERY            0
    DHCPBULKLEASEQUERY        0
    DHCPLEASEACTIVE           0
    DHCPLEASEUNASSIGNED       0
    DHCPLEASEUNKNOWN          0
```

```
    DHCPLEASEQUERYDONE          0
    DHCPACTIVELEASEQUERY        0

Packets forwarded:
    Total                       5
    BOOTREQUEST                 2
    BOOTREPLY                   3
```

The Core 2 device received the request and forwarded the reply and offer.

3. Verify DHCP is working on the server and the client. First, confirm the server has a binding for the client.

```
user@DHCP-ServerC> show dhcp server binding
IP address        Session Id  Hardware address   Expires     State      Interface
10.1.102.101      32          56:04:15:00:dd:f3  86058       BOUND      ge-0/0/0.0
```

The server has a binding matching the client's MAC.

4. Confirm the client has an IP address and route.

```
user@Client-ServerB> show route

inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Access-internal/12] 00:01:53, metric 0
                    >  to 10.1.102.254 via ge-0/0/3.0
10.1.102.0/24      *[Direct/0] 00:01:53
                    >  via ge-0/0/3.0
10.1.102.101/32    *[Local/0] 00:01:53
                       Local via ge-0/0/3.0
```

The client has an IP address and a default route.

5. Ping from the client to Server A for end-to-end verification.

```
user@Client-ServerB> show interfaces ge-0/0/3 | match Current
  Current address: 56:04:15:00:dd:f3, Hardware address: 56:04:15:00:dd:f3

user@Client-ServerB> ping 10.1.101.101 count 2
PING 10.1.101.101 (10.1.101.101): 56 data bytes
```

```
64 bytes from 10.1.101.101: icmp_seq=0 ttl=62 time=6.750 ms
64 bytes from 10.1.101.101: icmp_seq=1 ttl=62 time=7.476 ms

--- 10.1.101.101 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 6.750/7.113/7.476/0.363 ms
```

The client can ping Server A.

How to Configure Loop Protection

IN THIS SECTION

- Requirements | 60
- Overview | 60
- Configuration | 61

Requirements

You can configure loop protection on the following devices that you configured in the "How to Configure an EVPN-VXLAN Fabric for a Campus Network With CRB" on page 16 configuration example:

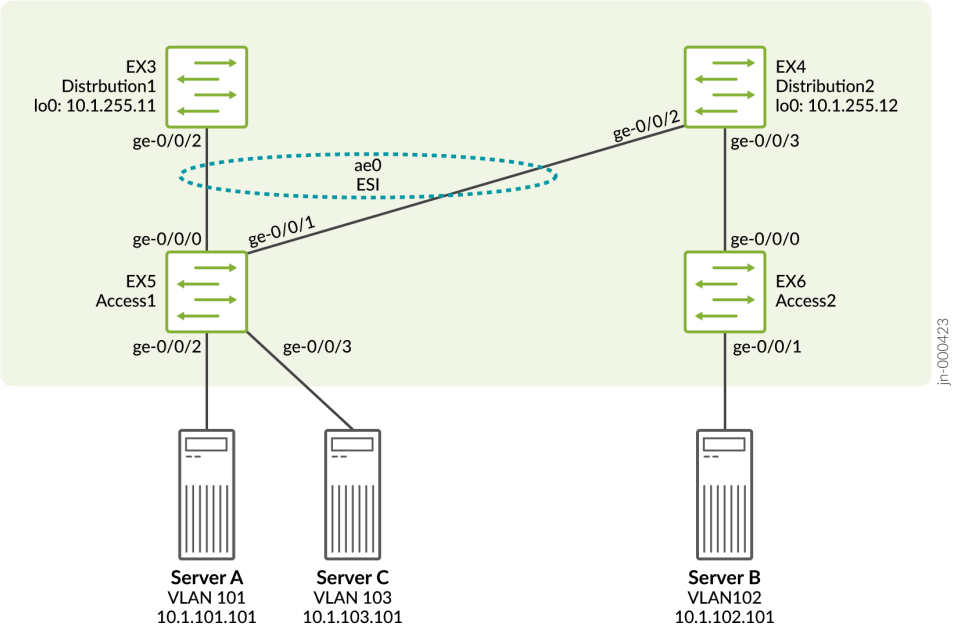
- Two EX4600 switch as the distribution devices. Software version: Junos OS Release 18.4R2-S4.5.
- Two access layer switches. This can be a Juniper Networks access switch or a third-party switch.
  - Re-validated using Junos OS Release 21.2R.3.
  - See the Feature Explorer for supported platforms.

Overview

EVPN protects the network against Layer 2 loops through split horizon, as described in RFC 7432. However, you might want to configure additional loop protection on your network. A Spanning Tree

Protocol (xSTP) prevents accidentally created loops between the access and distribution layers when a network administrator or user adds a new device to the network.

Figure 16: Access Layer Topology



Configuration

Use this section to configure loop protection on the distribution and access layers. We are only showing the Access 1 switch configuration. The distribution and the other access switch configuration is the same.

1. Configure the distribution and access switches.

   All Layer 2 ports on the distribution and access layers that are not part of EVPN ESIs must be configured under RSTP as edge ports. Enable the option to `bpdu-block-on-edge`.

```
user@Access1# set protocols rstp interface all edge
user@Access1# set protocols rstp bpdu-block-on-edge
```

SEE ALSO

Understanding Root Protection for STP, RSTP, and MSTP

No Link Title

Verify Loop Protection

Log in to the applicable devices to verify spanning tree is working and there are no blocked ports. Check on the distribution and access switches. We are only showing the outputs from Access 1.

1. Show the details for the spanning tree interfaces.

```
user@Access1> show spanning-tree interface

Spanning tree interface parameters for instance 0

Interface                 Port ID    Designated         Designated         Port    State
Role
                                      port ID           bridge ID          Cost
ae0                         128:3        128:3  32768.2c6bf5e794d0        10000    FWD
DESG
ge-0/0/2                  128:492      128:492  32768.2c6bf5e794d0        20000    FWD
DESG
ge-0/0/3                  128:493      128:493  32768.2c6bf5e794d0        20000    FWD
DESG
```

2. Show the details of the spanning tree bridge.

```
user@Access1> show spanning-tree bridge
STP bridge parameters
Routing instance name              : GLOBAL
Context ID                         : 0
Enabled protocol                   : RSTP
  Root ID                          : 32768.2c:6b:f5:e7:94:d0
  Hello time                       : 2 seconds
  Maximum age                      : 20 seconds
  Forward delay                    : 15 seconds
  Message age                      : 0
  Number of topology changes       : 1
  Time since last topology change  : 2723 seconds
```

```
Local parameters
  Bridge ID                       : 32768.2c:6b:f5:e7:94:d0
  Extended system ID              : 0
```

The outputs show Access 1 is forwarding on all ports and is a root bridge. There are no blocked ports.

RELATED DOCUMENTATION

- Overview of EX Series Switches and the Juniper Mist Cloud
- Collapsed Core with EVPN Multihoming

EVPN-VXLAN Campus Network Scaling Data for CRB

Table 1 on page 63 lists the scaling values for the EX9251 and EX4650 switches in the EVPN-VXLAN CRB campus example. You can find the topology and configuration in "How to Configure an EVPN-VXLAN Fabric for a Campus Network With CRB" on page 16.

These numbers do not represent the maximum supported scale. They represent only the solution testing performed for this example. The results of these tests are provided for reference purposes only.

Table 1: Scaling Values for CRB-Based EVPN-VXLAN Campus Network

Attribute	Fabric	Per Core Node (EX9251)	Distribution Node (EX4650)	Distribution Node (EX4400)
Virtual switches	1	1	1	1
VLANS/bridge domains/ VNIs	4000	4000	4000	4000
ESIs	62	Not Applicable	62	30
IRB interfaces	4000	4000	Not Applicable	Not Applicable

Table 1: Scaling Values for CRB-Based EVPN-VXLAN Campus Network *(Continued)*

Attribute	Fabric	Per Core Node (EX9251)	Distribution Node (EX4650)	Distribution Node (EX4400)
VRF instances	2000	2000	Not Applicable	Not Applicable
ARP entries	60,000	60,000[1]	Not Applicable	Not Applicable
ND scale (incudes link local)	60,000	60,000[1]	Not Applicable	Not Applicable
Multicast *,G	15,000	15,000	Not Applicable	Not Applicable
Multicast S,G	15,000	15,000	Not Applicable	Not Applicable
IGMP snooping	15,000	15,000	15,000	16,000
MAC addresses	60,000	60,000[2]	60,000[3]	120,000
Unique MAC address configured per IRB	100	100	Not Applicable	Not Applicable
Unique Virtual GW MAC address configured per IRB	100	100	Not Applicable	Not Applicable
DHCP-Relay Session	30,000	30,000	Not Applicable	Not Applicable

[1]Tested with 60,000 ARP and ND entries. The EX9251 switch can support up to 128,000 entries. The EX9253 switch can scale up to 256,000 ARP entries with EVPN-VXLAN.

[2]Tested with 60,000 MAC addresses on the EX9251 switch. We can scale up to 500,000 pure Layer 2 MAC Addresses.

[3]Tested with 60,000 MAC addresses on the EX4650 switch. We can scale up to 291,000 pure Layer 2 MAC addresses with Layer 2 profiles.

Appendix: Full Device Configurations

IN THIS SECTION

- Core Device Configurations | 65
- Distribution Device Configurations | 72
- Access Device Configurations | 75
- MX Series Router Configuration | 76
- SRX Series Device Configuration | 76

This appendix provides the baseline configuration for all devices in the example topology shown in Figure 9 on page 18. To quickly configure this topology, modify the configurations to match your network and copy them onto your devices.

NOTE: These configurations don't reflect optional features like DHCP. To to add the optional functionality, simply add the optional configuration to the baseline configurations below.

Core Device Configurations

IN THIS SECTION

- Core 1 Configuration | 65
- Core 2 Configuration | 68

Core 1 Configuration

```
set interfaces ge-0/0/0 description to_distribution1
set interfaces ge-0/0/0 mtu 9000
```

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.11.1/30
set interfaces ge-0/0/1 description to_distribution2
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 10.1.12.1/30
set interfaces ge-0/0/2 description to_mx
set interfaces ge-0/0/2 mtu 9000
set interfaces ge-0/0/2 unit 0 family inet address 10.1.112.1/30
set interfaces ge-0/0/4 vlan-tagging
set interfaces ge-0/0/4 mtu 9000
set interfaces ge-0/0/4 unit 101 description v101_to_srx
set interfaces ge-0/0/4 unit 101 vlan-id 101
set interfaces ge-0/0/4 unit 101 family inet address 10.1.111.1/30
set interfaces ge-0/0/4 unit 102 description v102_to_srx
set interfaces ge-0/0/4 unit 102 vlan-id 102
set interfaces ge-0/0/4 unit 102 family inet address 10.1.121.1/30
set interfaces irb mtu 9000
set interfaces irb unit 101 proxy-macip-advertisement
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.1/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 101 virtual-gateway-v4-mac 00:12:12:12:12:12
set interfaces irb unit 102 proxy-macip-advertisement
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.1/24 virtual-gateway-address
10.1.102.254
set interfaces irb unit 102 virtual-gateway-v4-mac 00:12:12:12:12:12
set interfaces irb unit 103 proxy-macip-advertisement
set interfaces irb unit 103 virtual-gateway-accept-data
set interfaces irb unit 103 family inet address 10.1.103.1/24 virtual-gateway-address
10.1.103.254
set interfaces irb unit 103 virtual-gateway-v4-mac 00:12:12:12:12:12
set interfaces lo0 unit 0 family inet address 10.1.255.1/32
set interfaces lo0 unit 101 family inet address 10.101.255.1/32
set interfaces lo0 unit 102 family inet address 10.102.255.1/32
set interfaces lo0 unit 103 family inet address 10.103.255.1/32
set policy-options policy-statement adv-vrfs term 1 from protocol static
set policy-options policy-statement adv-vrfs term 1 from route-filter 10.1.101.0/24 exact
set policy-options policy-statement adv-vrfs term 1 then accept
set policy-options policy-statement default-rt term 1 from route-filter 0.0.0.0/0 exact
set policy-options policy-statement default-rt term 1 then accept
set policy-options policy-statement default-rt term 2 then reject
set policy-options policy-statement lb_policy then load-balance per-packet
set policy-options policy-statement lb_policy then accept
```

```
set policy-options policy-statement serverB_imp term 1 from community serverC
set policy-options policy-statement serverB_imp term 1 from route-filter 10.1.103.0/24 exact
set policy-options policy-statement serverB_imp term 1 then accept
set policy-options policy-statement serverC_imp term 1 from community serverB
set policy-options policy-statement serverC_imp term 1 from route-filter 10.1.102.0/24 exact
set policy-options policy-statement serverC_imp term 1 then accept
set policy-options community serverB members target:65000:102
set policy-options community serverC members target:65000:103
set routing-instances vrf_101 instance-type vrf
set routing-instances vrf_101 routing-options router-id 10.101.255.1
set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface irb.101 passive
set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface lo0.101
set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface ge-0/0/4.101 interface-type
p2p
set routing-instances vrf_101 interface ge-0/0/4.101
set routing-instances vrf_101 interface irb.101
set routing-instances vrf_101 interface lo0.101
set routing-instances vrf_101 route-distinguisher 10.101.255.1:101
set routing-instances vrf_101 vrf-target target:65000:101
set routing-instances vrf_102 instance-type vrf
set routing-instances vrf_102 routing-options router-id 10.102.255.1
set routing-instances vrf_102 routing-options auto-export
set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface irb.102 passive
set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface lo0.102
set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface ge-0/0/4.102 interface-type
p2p
set routing-instances vrf_102 interface ge-0/0/4.102
set routing-instances vrf_102 interface irb.102
set routing-instances vrf_102 interface lo0.102
set routing-instances vrf_102 route-distinguisher 10.102.255.1:102
set routing-instances vrf_102 vrf-import serverB_imp
set routing-instances vrf_102 vrf-target target:65000:102
set routing-instances vrf_103 instance-type vrf
set routing-instances vrf_103 routing-options auto-export
set routing-instances vrf_103 interface irb.103
set routing-instances vrf_103 interface lo0.103
set routing-instances vrf_103 route-distinguisher 10.103.255.1:103
set routing-instances vrf_103 vrf-import serverC_imp
set routing-instances vrf_103 vrf-target target:65000:103
set routing-instances vs_1 instance-type virtual-switch
set routing-instances vs_1 protocols evpn encapsulation vxlan
set routing-instances vs_1 protocols evpn default-gateway no-gateway-community
set routing-instances vs_1 protocols evpn extended-vni-list all
```

```
set routing-instances vs_1 vtep-source-interface lo0.0
set routing-instances vs_1 route-distinguisher 10.1.255.1:1
set routing-instances vs_1 vrf-target target:65000:1
set routing-instances vs_1 vrf-target auto
set routing-instances vs_1 vlans v101 vlan-id 101
set routing-instances vs_1 vlans v101 l3-interface irb.101
set routing-instances vs_1 vlans v101 vxlan vni 1101
set routing-instances vs_1 vlans v101 vxlan ingress-node-replication
set routing-instances vs_1 vlans v102 vlan-id 102
set routing-instances vs_1 vlans v102 l3-interface irb.102
set routing-instances vs_1 vlans v102 vxlan vni 1102
set routing-instances vs_1 vlans v102 vxlan ingress-node-replication
set routing-instances vs_1 vlans v103 vlan-id 103
set routing-instances vs_1 vlans v103 l3-interface irb.103
set routing-instances vs_1 vlans v103 vxlan vni 1103
set routing-instances vs_1 vlans v103 vxlan ingress-node-replication
set routing-options router-id 10.1.255.1
set routing-options autonomous-system 65000
set routing-options static route 10.1.101.0/24 next-table vrf_101.inet.0
set routing-options rib-groups wan import-rib inet.0
set routing-options rib-groups wan import-rib vrf_101.inet.0
set routing-options rib-groups wan import-policy default-rt
set routing-options forwarding-table export lb_policy
set protocols bgp group OVERLAY type internal
set protocols bgp group OVERLAY local-address 10.1.255.1
set protocols bgp group OVERLAY family evpn signaling
set protocols bgp group OVERLAY cluster 10.1.255.255
set protocols bgp group OVERLAY multipath
set protocols bgp group OVERLAY neighbor 10.1.255.11
set protocols bgp group OVERLAY neighbor 10.1.255.12
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf rib-groups inet wan
set protocols ospf export adv-vrfs
```

Core 2 Configuration

```
set interfaces ge-0/0/0 description to_distribution1
set interfaces ge-0/0/0 mtu 9000
```

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.21.1/30
set interfaces ge-0/0/1 description to_distribution2
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 10.1.22.1/30
set interfaces ge-0/0/2 description to_mx
set interfaces ge-0/0/2 mtu 9000
set interfaces ge-0/0/2 unit 0 family inet address 10.1.212.1/30
set interfaces ge-0/0/3 vlan-tagging
set interfaces ge-0/0/3 mtu 9000
set interfaces ge-0/0/3 unit 101 description v101_to_srx
set interfaces ge-0/0/3 unit 101 vlan-id 101
set interfaces ge-0/0/3 unit 101 family inet address 10.1.211.1/30
set interfaces ge-0/0/3 unit 102 description v102_to_srx
set interfaces ge-0/0/3 unit 102 vlan-id 102
set interfaces ge-0/0/3 unit 102 family inet address 10.1.221.1/30
set interfaces irb mtu 9000
set interfaces irb unit 101 proxy-macip-advertisement
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.2/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 101 virtual-gateway-v4-mac 00:12:12:12:12:12
set interfaces irb unit 102 proxy-macip-advertisement
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.2/24 virtual-gateway-address
10.1.102.254
set interfaces irb unit 102 virtual-gateway-v4-mac 00:12:12:12:12:12
set interfaces irb unit 103 proxy-macip-advertisement
set interfaces irb unit 103 virtual-gateway-accept-data
set interfaces irb unit 103 family inet address 10.1.103.2/24 virtual-gateway-address
10.1.103.254
set interfaces irb unit 103 virtual-gateway-v4-mac 00:12:12:12:12:12
set interfaces lo0 unit 0 family inet address 10.1.255.2/32
set interfaces lo0 unit 101 family inet address 10.101.255.2/32
set interfaces lo0 unit 102 family inet address 10.102.255.2/32
set interfaces lo0 unit 103 family inet address 10.103.255.2/32
set policy-options policy-statement adv-vrfs term 1 from protocol static
set policy-options policy-statement adv-vrfs term 1 from route-filter 10.1.101.0/24 exact
set policy-options policy-statement adv-vrfs term 1 then accept
set policy-options policy-statement default-rt term 1 from route-filter 0.0.0.0/0 exact
set policy-options policy-statement default-rt term 1 then accept
set policy-options policy-statement default-rt term 2 then reject
set policy-options policy-statement lb_policy then load-balance per-packet
set policy-options policy-statement lb_policy then accept
```

```
set policy-options policy-statement serverB_imp term 1 from community serverC
set policy-options policy-statement serverB_imp term 1 from route-filter 10.1.103.0/24 exact
set policy-options policy-statement serverB_imp term 1 then accept
set policy-options policy-statement serverC_imp term 1 from community serverB
set policy-options policy-statement serverC_imp term 1 from route-filter 10.1.102.0/24 exact
set policy-options policy-statement serverC_imp term 1 then accept
set policy-options community serverB members target:65000:102
set policy-options community serverC members target:65000:103
set routing-instances vrf_101 instance-type vrf
set routing-instances vrf_101 routing-options router-id 10.101.255.2
set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface irb.101 passive
set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface lo0.101
set routing-instances vrf_101 protocols ospf area 0.0.0.0 interface ge-0/0/3.101 interface-type
p2p
set routing-instances vrf_101 interface ge-0/0/3.101
set routing-instances vrf_101 interface irb.101
set routing-instances vrf_101 interface lo0.101
set routing-instances vrf_101 route-distinguisher 10.101.255.2:101
set routing-instances vrf_101 vrf-target target:65000:101
set routing-instances vrf_102 instance-type vrf
set routing-instances vrf_102 routing-options router-id 10.102.255.2
set routing-instances vrf_102 routing-options auto-export
set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface irb.102 passive
set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface lo0.102
set routing-instances vrf_102 protocols ospf area 0.0.0.0 interface ge-0/0/3.102 interface-type
p2p
set routing-instances vrf_102 interface ge-0/0/3.102
set routing-instances vrf_102 interface irb.102
set routing-instances vrf_102 interface lo0.102
set routing-instances vrf_102 route-distinguisher 10.102.255.2:102
set routing-instances vrf_102 vrf-import serverB_imp
set routing-instances vrf_102 vrf-target target:65000:102
set routing-instances vrf_103 instance-type vrf
set routing-instances vrf_103 routing-options auto-export
set routing-instances vrf_103 interface irb.103
set routing-instances vrf_103 interface lo0.103
set routing-instances vrf_103 route-distinguisher 10.103.255.2:103
set routing-instances vrf_103 vrf-import serverC_imp
set routing-instances vrf_103 vrf-target target:65000:103
set routing-instances vs_1 instance-type virtual-switch
set routing-instances vs_1 protocols evpn encapsulation vxlan
set routing-instances vs_1 protocols evpn default-gateway no-gateway-community
set routing-instances vs_1 protocols evpn extended-vni-list all
```

```
set routing-instances vs_1 vtep-source-interface lo0.0
set routing-instances vs_1 route-distinguisher 10.1.255.2:1
set routing-instances vs_1 vrf-target target:65000:1
set routing-instances vs_1 vrf-target auto
set routing-instances vs_1 vlans v101 vlan-id 101
set routing-instances vs_1 vlans v101 l3-interface irb.101
set routing-instances vs_1 vlans v101 vxlan vni 1101
set routing-instances vs_1 vlans v101 vxlan ingress-node-replication
set routing-instances vs_1 vlans v102 vlan-id 102
set routing-instances vs_1 vlans v102 l3-interface irb.102
set routing-instances vs_1 vlans v102 vxlan vni 1102
set routing-instances vs_1 vlans v102 vxlan ingress-node-replication
set routing-instances vs_1 vlans v103 vlan-id 103
set routing-instances vs_1 vlans v103 l3-interface irb.103
set routing-instances vs_1 vlans v103 vxlan vni 1103
set routing-instances vs_1 vlans v103 vxlan ingress-node-replication
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 65000
set routing-options static route 10.1.101.0/24 next-table vrf_101.inet.0
set routing-options rib-groups wan import-rib inet.0
set routing-options rib-groups wan import-rib vrf_101.inet.0
set routing-options rib-groups wan import-policy default-rt
set routing-options forwarding-table export lb_policy
set protocols bgp group OVERLAY type internal
set protocols bgp group OVERLAY local-address 10.1.255.2
set protocols bgp group OVERLAY family evpn signaling
set protocols bgp group OVERLAY cluster 10.1.255.255
set protocols bgp group OVERLAY multipath
set protocols bgp group OVERLAY neighbor 10.1.255.11
set protocols bgp group OVERLAY neighbor 10.1.255.12
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf rib-groups inet wan
set protocols ospf export adv-vrfs
```

Distribution Device Configurations

IN THIS SECTION

- Distribution 1 Configuration | 72
- Distribution 2 Configuration | 73

Distribution 1 Configuration

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 description to_core1
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 10.1.11.2/30
set interfaces ge-0/0/1 description to_core2
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 10.1.21.2/30
set interfaces ge-0/0/2 description to_access1
set interfaces ge-0/0/2 ether-options 802.3ad ae0
set interfaces ae0 mtu 9000
set interfaces ae0 esi 00:00:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:01:01:01:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members v101
set interfaces ae0 unit 0 family ethernet-switching vlan members v102
set interfaces ae0 unit 0 family ethernet-switching vlan members v103
set interfaces lo0 unit 0 family inet address 10.1.255.11/32
set policy-options policy-statement lb_policy then load-balance per-packet
set policy-options policy-statement lb_policy then accept
set routing-options router-id 10.1.255.11
set routing-options autonomous-system 65000
set routing-options forwarding-table export lb_policy
set protocols bgp group OVERLAY type internal
set protocols bgp group OVERLAY local-address 10.1.255.11
set protocols bgp group OVERLAY family evpn signaling
```

```
set protocols bgp group OVERLAY multipath
set protocols bgp group OVERLAY neighbor 10.1.255.1
set protocols bgp group OVERLAY neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn extended-vni-list 1101
set protocols evpn extended-vni-list 1102
set protocols evpn extended-vni-list 1103
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.11:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 vxlan vni 1101
set vlans v101 vxlan ingress-node-replication
set vlans v102 vlan-id 102
set vlans v102 vxlan vni 1102
set vlans v102 vxlan ingress-node-replication
set vlans v103 vlan-id 103
set vlans v103 vxlan vni 1103
set vlans v103 vxlan ingress-node-replication
```

Distribution 2 Configuration

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 description to_core1
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 10.1.12.2/30
set interfaces ge-0/0/1 description to_core2
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 10.1.22.2/30
set interfaces ge-0/0/2 description to_access1
set interfaces ge-0/0/2 ether-options 802.3ad ae0
set interfaces ge-0/0/3 description to_access2
set interfaces ge-0/0/3 mtu 9000
set interfaces ge-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members v101
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members v102
```

```
set interfaces ae0 mtu 9000
set interfaces ae0 esi 00:00:01:01:01:01:01:01:01:01
set interfaces ae0 esi all-active
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 aggregated-ether-options lacp system-id 00:00:01:01:01:01
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members v101
set interfaces ae0 unit 0 family ethernet-switching vlan members v102
set interfaces ae0 unit 0 family ethernet-switching vlan members v103
set interfaces lo0 unit 0 family inet address 10.1.255.12/32
set policy-options policy-statement lb_policy then load-balance per-packet
set policy-options policy-statement lb_policy then accept
set routing-options router-id 10.1.255.12
set routing-options autonomous-system 65000
set routing-options forwarding-table export lb_policy
set protocols bgp group OVERLAY type internal
set protocols bgp group OVERLAY local-address 10.1.255.12
set protocols bgp group OVERLAY family evpn signaling
set protocols bgp group OVERLAY multipath
set protocols bgp group OVERLAY neighbor 10.1.255.1
set protocols bgp group OVERLAY neighbor 10.1.255.2
set protocols evpn encapsulation vxlan
set protocols evpn multicast-mode ingress-replication
set protocols evpn extended-vni-list 1101
set protocols evpn extended-vni-list 1102
set protocols evpn extended-vni-list 1103
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.12:1
set switch-options vrf-target target:65000:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 vxlan vni 1101
set vlans v101 vxlan ingress-node-replication
set vlans v102 vlan-id 102
set vlans v102 vxlan vni 1102
set vlans v102 vxlan ingress-node-replication
set vlans v103 vlan-id 103
```

```
set vlans v103 vxlan vni 1103
set vlans v103 vxlan ingress-node-replication
```

Access Device Configurations

IN THIS SECTION

- Access 1 Configuration | 75
- Access 2 Configuration | 76

Access 1 Configuration

```
set chassis aggregated-devices ethernet device-count 1
set interfaces ge-0/0/0 description to_dist1
set interfaces ge-0/0/0 ether-options 802.3ad ae0
set interfaces ge-0/0/1 description to_dist2
set interfaces ge-0/0/1 ether-options 802.3ad ae0
set interfaces ge-0/0/2 mtu 9000
set interfaces ge-0/0/2 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/2 unit 0 family ethernet-switching vlan members v101
set interfaces ge-0/0/3 mtu 9000
set interfaces ge-0/0/3 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/3 unit 0 family ethernet-switching vlan members v103
set interfaces ae0 mtu 9000
set interfaces ae0 aggregated-ether-options minimum-links 1
set interfaces ae0 aggregated-ether-options lacp active
set interfaces ae0 aggregated-ether-options lacp periodic fast
set interfaces ae0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ae0 unit 0 family ethernet-switching vlan members v101
set interfaces ae0 unit 0 family ethernet-switching vlan members v103
set vlans v101 vlan-id 101
set vlans v103 vlan-id 103
```

Access 2 Configuration

```
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family ethernet-switching interface-mode trunk
set interfaces ge-0/0/0 unit 0 family ethernet-switching vlan members v102
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family ethernet-switching interface-mode access
set interfaces ge-0/0/1 unit 0 family ethernet-switching vlan members v102
set vlans v102 vlan-id 102
```

MX Series Router Configuration

```
set interfaces ge-0/0/0 description to_core1
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 0 family inet address 10.1.112.2/30
set interfaces ge-0/0/1 description to_core2
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 0 family inet address 10.1.212.2/30
set interfaces ge-0/0/2 mtu 9000
set interfaces ge-0/0/2 unit 0 family inet address 172.16.1.1/30
set interfaces lo0 unit 0 family inet address 10.1.255.112/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 interface-type p2p
set protocols ospf area 0.0.0.0 interface lo0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 passive
set protocols ospf export adv-stat
set policy-options policy-statement adv-stat term 1 from protocol static
set policy-options policy-statement adv-stat term 1 then accept
set routing-options router-id 10.1.255.112
set routing-options static route 0.0.0.0/0 next-hop 172.16.1.2
```

SRX Series Device Configuration

```
set security address-book global address v101 10.1.101.0/24
set security address-book global address v102 10.1.102.0/24
```

```
set security policies from-zone v101 to-zone v102 policy v101_to_v102 match source-address v101
set security policies from-zone v101 to-zone v102 policy v101_to_v102 match destination-address
v102
set security policies from-zone v101 to-zone v102 policy v101_to_v102 match application any
set security policies from-zone v101 to-zone v102 policy v101_to_v102 then permit
set security policies from-zone v102 to-zone v101 policy v102_to_v101 match source-address v102
set security policies from-zone v102 to-zone v101 policy v102_to_v101 match destination-address
v101
set security policies from-zone v102 to-zone v101 policy v102_to_v101 match application any
set security policies from-zone v102 to-zone v101 policy v102_to_v101 then permit
set security zones security-zone v101 host-inbound-traffic system-services ping
set security zones security-zone v101 host-inbound-traffic protocols ospf
set security zones security-zone v101 interfaces ge-0/0/0.101
set security zones security-zone v101 interfaces ge-0/0/1.101
set security zones security-zone v102 host-inbound-traffic system-services ping
set security zones security-zone v102 host-inbound-traffic protocols ospf
set security zones security-zone v102 interfaces ge-0/0/1.102
set security zones security-zone v102 interfaces ge-0/0/0.102
set interfaces ge-0/0/0 description to_core1
set interfaces ge-0/0/0 vlan-tagging
set interfaces ge-0/0/0 mtu 9000
set interfaces ge-0/0/0 unit 101 vlan-id 101
set interfaces ge-0/0/0 unit 101 family inet address 10.1.111.2/30
set interfaces ge-0/0/0 unit 102 vlan-id 102
set interfaces ge-0/0/0 unit 102 family inet address 10.1.121.2/30
set interfaces ge-0/0/1 description to_core2
set interfaces ge-0/0/1 vlan-tagging
set interfaces ge-0/0/1 mtu 9000
set interfaces ge-0/0/1 unit 101 vlan-id 101
set interfaces ge-0/0/1 unit 101 family inet address 10.1.211.2/30
set interfaces ge-0/0/1 unit 102 vlan-id 102
set interfaces ge-0/0/1 unit 102 family inet address 10.1.221.2/30
set interfaces lo0 unit 1 family inet address 10.1.255.111/32
set routing-instances v101_v102 instance-type virtual-router
set routing-instances v101_v102 routing-options router-id 10.1.255.111
set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/0.101 interface-
type p2p
set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/0.102 interface-
type p2p
set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/1.101 interface-
type p2p
set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface ge-0/0/1.102 interface-
type p2p
```

```
set routing-instances v101_v102 protocols ospf area 0.0.0.0 interface lo0.1
set routing-instances v101_v102 interface ge-0/0/0.101
set routing-instances v101_v102 interface ge-0/0/0.102
set routing-instances v101_v102 interface ge-0/0/1.101
set routing-instances v101_v102 interface ge-0/0/1.102
set routing-instances v101_v102 interface lo0.1
```